COLLEGES SCIENCES
BRETAGNE POUR L'INGENIEUR
LOIRE ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THE UNIVERSITY of ADELAIDE

THÈSE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPÉRIEURE MINES-TÉLÉCOM ATLANTIQUE BRETAGNE PAYS DE LA LOIRE - IMT ATLANTIQUE

EN COTUTELLE AVEC

L'UNIVERSITÉ D'ADELAÏDE - AUSTRALIE MERIDIONALE

ÉCOLE DOCTORALE N° 648
Sciences pour l'Ingénieur et le Numérique
Spécialité : *Informatique*

Par
Raphaël LAFARGUE

Few-Shot Learning : A Data-centric Approach for Adaptation

Thèse présentée et soutenue à IMT Atlantique, Brest, le 4 Décembre 2024
Unité de recherche : Lab-STICC UMR 6285
Thèse N° : 2024IMTA0439

Rapporteurs avant soutenance :

Céline HUDELOT Professeur, CentraleSupelec
Denis PELERIN Professeur, Université Grenoble Alpes

Composition du Jury :

Président :	Nicolas COURTY	Professeur, Université Bretagne Sud
Examinateurs :	Denis PELERIN	Professeur, Université Grenoble Alpes
	Céline HUDELOT	Professeur, CentraleSupelec
Dir. de thèse :	Vincent GRIPON	Directeur de Recherche, IMT Atlantique
Co-dir. de thèse :	Ian REID	Professeur à University of Adelaide

Acknowledgement

After three years of hard work, it is time to express my heartfelt gratitude to the incredible people who supported me along the way. Meeting and getting to know all of you has been a true blessing. Thank you!

First and foremost, I am deeply grateful to Vincent for guiding me throughout this journey. Despite managing two newborns and many students across the globe, you were always there to ensure my projects stayed on track and to provide invaluable support. Your dedication has been truly inspiring, and I cannot thank you enough.

I extend my sincere thanks to Jack and Ian, whose contributions gave this PhD a renewed momentum when it was needed the most. Jack, I greatly appreciate your hard work and dedication to our projects—it was a privilege to have you on board, and I will always cherish the fantastic experiences in Adelaide. Ian, thank you for the insightful discussions and the valuable contributions you brought to this journey.

To all my supervisors, thank you for being such excellent mentors.

My heartfelt thanks also go to the jury members—Nicolas Courty, Céline Hudelot, and Denis Pellerin—for their thorough review and thoughtful evaluation of my PhD.

I am also grateful to Jean-Philippe Diguet. Your help was instrumental in making the Franco-Australian collaboration a reality, and I truly appreciated our scientific discussions at IRL Crossing.

Many thanks to my co-authors and collaborators—Yassir Bendou, Nilesh Ramgolam, Luke Smith, Giulia Lioi, Bastien Pasdeloup, Matthias Lowe, Franck Vermet, and many others. Working with you was both professionally enriching and personally rewarding. It was also fun and really enjoyable.

I am profoundly thankful to my friends and family who shared this journey with me. Your presence in both the highs and lows has meant the world to me. These three years were filled with joyful moments, and the time spent in Brittany and South Australia ranks among the best of my life. I miss you all dearly.

To my friends from Brest: Reda, Ilyass, Lucas B., Lucas G., Hugo L., Hugo T., Yassine, Amine, Manon, Jérémy, Thimothée, Albane, Alix, Brahim, Ismail, Nicolas, Axel, Mathieu, Gilles, and many others, thank you for the unforgettable memories. Special thanks to

Joachim, Antoine, Martin, and Katell for being there at my defense and for all the good times we shared.

To my friends from Australia: CK, Georgia, Chris Fu., Shin-Fang, Angel, Khalil, Nadhir, Chad, Nadia, Khan, Lana, Luke, Hélène, Quentin, Katell, Maëlic, Antoine, Camille, Quyen, Chris Fo., Thomas R., Thomas C., Loretta, Alexandre, Cédric, Benoit, Adrien, Océane, Jonno, David, Alejandro, Charlie, Pavan, Laura, and so many others—thank you for making the year and a half I spent with you truly unforgettable. I loved all the trips and adventures we went on together.

Finally, I owe my deepest gratitude to my parents and my brother for shaping me into who I am today and for their unwavering support in all my endeavors.

Résumé en Français

Introduction

L'apprentissage profond (Deep Learning) a connu une progression spectaculaire au cours des quinze dernières années. Des avancées majeures, partant de la reconnaissance de chiffres manuscrits [1] à la génération de vidéos ou la prédiction de structures protéiques [2], ont été possibles grâce à l'abondance de données et à l'amélioration de la puissance de calcul. Récemment, ces progrès ont donné naissance aux modèles fondation, comme les transformateurs visuels (Vision Transformers, ViT) [3], qui sont devenus des outils universels capables de traiter des tâches variées en s'appuyant sur des ensembles de données massifs pour leur pré-entraînement.

Cependant, cette ère des modèles à grande échelle soulève plusieurs défis, notamment en matière de confidentialité des données, d'explicabilité (XAI), et de capacité de généralisation. Ce dernier aspect, qui est au cœur de cette thèse, questionne la capacité des modèles à s'adapter efficacement à des tâches nouvelles et à des domaines inédits, surtout lorsqu'ils ne disposent que de peu de données.

L'*Apprentissage Parcimonieux* (AP) répond à ce besoin en explorant comment les modèles peuvent apprendre de nouvelles tâches à partir de très peu d'exemples, une capacité où les humains excellent. Nous nous concentrons sur la classification d'image dans cette thèse. Dans ce contexte, le pré-entraînement sur des ensembles de données volumineux joue un rôle crucial en réduisant les risques de sur-apprentissage sur le peu de données disponible. Cependant, une observation clé guide cette thèse : certaines données de pré-entraînement peuvent avoir un effet nuisible sur la performance en AP, en introduisant des biais ou des corrélations non pertinentes.

La problématique centrale de la thèse est donc formulée ainsi : *dans le cadre de l'Apprentissage Parcimonieux, comment identifier et atténuer l'impact des échantillons adverses dans les ensembles de données de pré-entraînement afin d'améliorer la généralisation des modèles ?*

Pour répondre à cette question, la thèse propose :

1. De développer des modèles robustes capables d'extraire des caractéristiques uni-

verselles adaptées au FSL.

2. De proposer de nouvelles méthodologies d'évaluation pour prendre en compte la variabilité introduite par les données.
3. D'optimiser l'utilisation des données de pré-entraînement en mettant en lumière l'impact des sous-domaines pertinents et nuisibles sur la généralisation.

Les principales contributions seront détaillées dans les chapitres suivants. D'un point de vue chronologique, ces contributions ont été réalisées dans l'ordre suivant : (1), (3), puis (2) La contribution (2), étant la plus récente, repose sur l'utilisation de modèles de fondation, ce qui n'était pas encore une norme au moment où les contributions (1) et (3) ont été développées.

Entraîner un Extracteur de Caractéristiques Robuste

Ce chapitre s'inspire grandement du papier *EASY: Ensemble Augmented-Shot Y-shaped Learning: State-Of-The-Art Few-Shot Classification with Simple Ingredients*, reçu comme meilleur papier de 2022 du *Journal of Imaging* [4].

Contexte et Objectifs

Il explore les moyens de concevoir des modèles capables de produire des représentations robustes et adaptées à l'Apprentissage Parcimonieux (AP). L'objectif est de maximiser les performances sur des tâches où peu de données sont disponibles, tout en simplifiant et en optimisant le processus d'entraînement. En effet, les méthodes présentées dans la littérature proposent souvent des approches qui ne sont pas basées sur un même pré-entraînement de référence ce qui complique l'interprétation des phénomène. Les gains annoncées sont-ils reproductible avec un meilleur pré-entraînement. Nous proposons donc d'optimiser le pré-entraînement avec des méthode simple pour obtenir une référence de qualité.

Pour s'assurer de la réelle nouveauté des données d'AP, nous utilisons le type de benchmark qui était prédominant au début de cette thèse [5, 6]. Il s'agit d'une séparation strict des données de pré-entraînement et des données.

Le jeu de données de pré-entraînement est l'ensemble de données contenant un grand nombre de classes et d'exemples variés. Le jeu de données nouveau est l'ensemble de

données contient des classes totalement distinctes de celles du jeu de données de pré-entraînement et sert à évaluer les performances des modèles sur des tâches échantillonnées. On utilise la notation K-way-S-shot-Q-requêtes (queries), où K désigne le nombre de classes, S le nombre d'exemples étiquetés par classe, et Q le nombre de requêtes (exemples à classer). La précision est mesurée par le taux de requêtes correctement classifiées.

Méthode

Ce chapitre introduit une méthode simple et efficace pour pré-entraîner un extracteur de caractéristiques robuste qui intègre plusieurs techniques trouvées communément dans la littérature de l'époque de la publication du papier.

— Augmentation des données : Génération d'exemples synthétiques pour enrichir les ensembles d'entraînement. En particulier, nous proposons de faire de l'augmentation de données par rognage (cropping) sur l'ensemble support ce qui est moins commun que de le faire lors du pré-entraînement (nous le faisons aussi). Nous proposons de moyenner les représentations des images rognées pour avoir une représentation plus robuste.

— Ensembles de modèles (Ensemble Learning) : Utilisation de plusieurs extracteur de caractéristique pour capturer des perspectives diverses sur les données. Les caractéristiques sont concaténées.

— Fonction de coût auto-supervisée : Combinaison de tâches supervisées et auto-supervisées pour améliorer la qualité des représentations apprises. Notre fonction de coût essaie de prédire quelle rotation a été appliquée parmi 0°, 90°, 180° , 270°.

— *Post-Processing*: Centrage et projection sur la sphère unité des caractéristiques extraites. Ceci permet de mieux séparer les classes car la fonction coût entropie croisée ne prend pas en compte la magnitude mais seulement les direction dans lesquelles les représentation sont positionnées dans l'espace latent.

On représente la dite méthode dans la Figure 1.

Résultats

Pour chaque tâche d'AP *inductive*, nous utilisons le plus proche centroïde (Nearest Class Mean, NCM) pour classifier les requêtes. Le terme *inductif* signifie que nous pouvons classifier une seule requête à la fois, contrairement au mode *transductif* où toutes les KQ requêtes sont disponibles pour aider à la classification. Dans ce cas, nous pro-

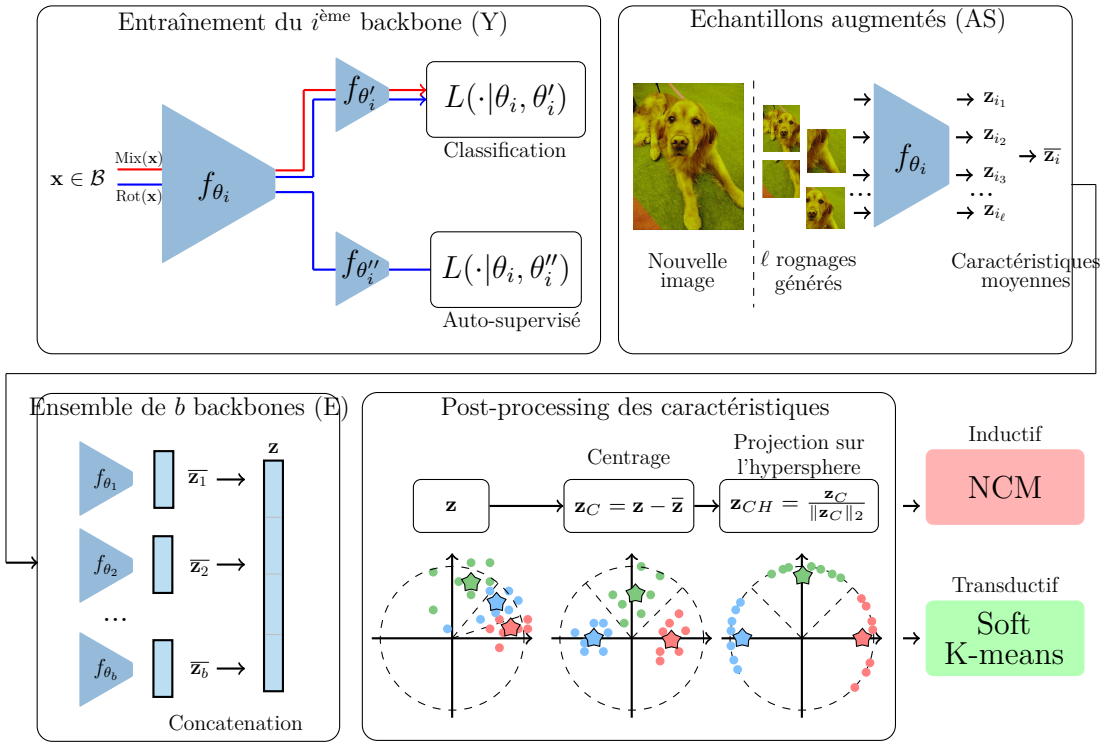


Figure 1 – Illustration de la méthode proposé dans [4] (CC-BY)

posons d'utiliser le *soft k-means*, une méthode inspirée du k-means classique, qui améliore itérativement les résultats du NCM.

La méthode atteint des performances à l'état de l'art en 1-shot 5-ways et 5-way-5-shot sur MiniImagneNet, TieredImageNet, CUB-FS, CIFAR-FS. Ces benchmarks étaient et sont pour certains toujours des benchmarks de référence dans le domaine. L'étude d'ablation des contributions individuelles montre que chaque ingrédient apporte un gain significatif à la précision mesurée. En résumé, les résultats expérimentaux montrent que cette approche dépasse des méthodes beaucoup plus complexes, établissant une nouvelle méthode à l'état de l'art à l'époque de sa publication et ce en inductif et en transductif.

Contribution de ce chapitre

Ce travail démontre que des ingrédients simples suffisent à atteindre des performances de pointe sur plusieurs benchmarks standards. En outre, il met en évidence les faiblesses des approches sophistiquées qui reposent souvent sur des pré-entraînement sous-optimaux en guise de point de départ.

Ce chapitre pose les fondations pour l'évaluation et l'amélioration des méthodologies en AP, explorées dans les chapitres suivants.

Réinterprétation des Intervalles de Confiance en Apprentissage Parcimonieux

Identification d'un Problème

Ce chapitre s'inspire largement de l'article *Oops I Sampled It Again: Reinterpreting Confidence Intervals in Few-Shot Learning*, publié dans TMLR en 2024 [7]. Il aborde le problème crucial de l'évaluation fiable des méthodes en Apprentissage Parcimonieux (AP). Nous démontrons notamment que la méthode d'évaluation peut avoir des effets déterminants sur les conclusions relatives à la supériorité d'une méthode d'AP par rapport à une autre.

Les ensembles support (*support set*) contenant les *shots* (données d'entraînement pour une tâche d'AP) étant de petite taille, ils engendrent une grande variabilité dans les précisions (*accuracy*) mesurées pour chaque tâche. Certaines tâches sont difficile (faible précision) et d'autres faciles (haute précision). Afin d'évaluer de manière équitable les

méthodes d'AP, un grand nombre de tâches est échantillonné, et la moyenne des précisions sur ces tâches est reportée.

Les évaluations actuelles en AP s'appuient souvent sur des intervalles de confiance (CI) pour quantifier l'incertitude des performances des modèles sur des tâches simulées. L'intervalle de confiance pour T tâches est généralement calculé comme suit :

$$CI_{95\%} = 1.96\frac{\sigma_A}{\sqrt{T}} \tag{1}$$

où σ_A représente l'écart-type des précisions mesurées.

Cependant, ces CI sont habituellement calculés en supposant que les tâches d'évaluation sont indépendantes, une hypothèse qui est rarement respectée en pratique. En effet, les tâches sont échantillonnées avec remise (une image peut apparaître dans plusieurs tâches).

Pour répondre à cette limitation, ce chapitre introduit une nouvelle distinction entre deux types d'intervalles de confiance :

— **Intervalles Fermés (ICF)** : Correspondent à la méthode prédominante dans la littérature. Calculés en échantillonnant avec remplacement, ces intervalles ignorent souvent les dépendances entre tâches, ce qui peut fausser les conclusions sur les performances relatives des modèles. Le nombre de tâches T est généralement fixé à 600 [8], mais peut atteindre 2000 [9], rendant ainsi les intervalles présentés artificiellement plus petits. Les conclusions tirées d'une analyse avec ICF ne sont valables qu'à *jeu de données* fixe et ne s'étendent pas à la distribution contrairement à l'ICO.

— **Intervalles Ouverts (ICO)** : Les tâches sont échantillonnées sans remise, (chaque image n'est échantillonnée que pour une seule tâche), ces intervalles tiennent compte de l'aléa des données et de l'échantillonnage des tâches. Le nombre de tâches est alors limité par la taille du jeu de données. En pratique, les tâches sont échantillonnées jusqu'à ce qu'il n'y ait plus assez d'échantillons pour former une tâche. Lorsque le nombre de tâches est particulièrement faible, la loi de Student est utilisée pour calculer l'ICO à la place de la formule de l'Équation 1.

Les ICF et ICO diffèrent significativement en taille, comme indiqué dans le Tableau 1. En moyenne, les ICO sont 3,8 fois plus larges que les ICF, révélant une incertitude sous-estimée par les intervalles fermés. Cela réduit également le nombre de comparaisons significatives (non-superposition des ICs) entre méthodes d'AP.

Jeu de données	Nombre d'images	Modèle / Echantillonnage / Méthode	CLIP Avec Remise	CLIP Sans Remise	DINO Avec Remise	DINO Sans Remise
DTD	840	LR	79.59 ± 0.52	84.00 ± 4.50	83.29 ± 0.51	86.10 ± 4.66
		FT	76.87 ± 0.56	80.76 ± 5.60	81.82 ± 0.51	84.19 ± 5.76
Quickdraw	7,710,295	LR	75.79 ± 0.66	75.54 ± 0.36	74.54 ± 0.68	74.07 ± 0.38
		FT	76.18 ± 0.69	75.93 ± 0.38	74.10 ± 0.70	73.61 ± 0.39

Table 1 – Comparaison des OCIs (sans rmise) et des CIF (avec remise) en 5-shots 5-ways et 15 requêtes (queries). CLIP et DINO sont deux modèles pré-entraînés. LR et FT correspondent respectivement à la Régression Logistique et à l'ajustement fin (finetuning). (CC-BY)

Méthode pour mitiger le problème

Pour rendre les comparaisons plus significatives, nous proposons deux approches. La première consiste à utiliser des tests par paire (Paired Tests), et la seconde à choisir judicieusement le nombre de requêtes (queries) par tâche, que nous appellerons le dimensionnement.

Le test par paire (PT) consiste à évaluer deux méthodes sur exactement les mêmes tâches et à rapporter la différence moyenne. La différence des moyennes équivaut à la moyenne des différences ; cependant, la variance des différences ne correspond pas à la différence des variances. Nous pouvons ainsi tirer parti du fait que les précisions (accuracies) des tâches sont fortement corrélées d'une méthode à l'autre. En d'autres termes, une tâche "facile" pour une méthode a de fortes chances de l'être également pour une autre, et inversement. Considérons un cas particulier et comparons les performances de deux méthodes, notées A et B (détails dans le manuscrit). Par exemple, si l'on compare A à B sur le jeu de données *Traffic Signs* en configuration 5-shots 5-ways, nous trouvons, en utilisant la méthode prédominante (CIF), que A sous-performe par rapport à B. Cette conclusion devient toutefois non significative si l'on interdit la remise dans l'échantillonnage des tâches (CIO). Il est surprenant de constater qu'en utilisant un test par paire, la conclusion est inversée de manière significative : A sur-performe B lorsque l'aléa des données est pris en compte.

Le dimensionnement des tâches joue également un rôle clé pour minimiser la taille des CIOs. En effet, si nous choisissons de mesurer une précision sur un faible nombre de requêtes, nous pouvons échantillonner davantage de tâches, car nous utilisons moins de données. Le prix à payer dans ce cas est une plus grande variabilité des précisions (la précision sur une seule requête peut varier entre 0% et 100%). Il existe un optimum

permettant de minimiser l'ICO. À travers un développement mathématique, des données simulées et des données réelles, nous démontrons l'existence de ce minimum ainsi que des moyens de l'identifier en fonction de Q, le nombre de requêtes.

La Figure 2 illustre que ces deux approches permettent d'augmenter le nombre de comparaisons significatives (ou concluantes). Nous avons donc proposé un benchmark exploitant ces deux méthodes.

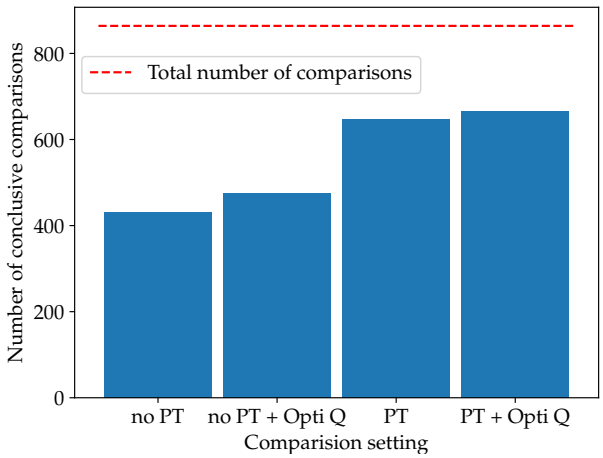


Figure 2 – Effet de l'utilisation de tests par paire et du dimensionnement de tâche sur le nombre de comparaisons concluantes avec 1, 5 et 10 shots. L'axe des ordonnées représente le nombre de comparaisons conclusives. Quand Q n'est pas optimisé, on choisit $Q = 15$. (CC-BY)

Comment se concentrer sur moins de données de pré-entraînement peut améliorer les performances en apprentissage parcimonieux ?

Ce chapitre est basé sur le pre-print *Few and Fewer: Learning Better from Few Examples Using Fewer Base Classes* [10] et explore une hypothèse intrigante : toutes les données utilisées pour le pré-entraînement ne contribuent pas également à la généralisation des modèles.

Contexte et Objectifs

Certaines parties des ensembles de données de pré-entraînement peuvent même avoir un impact négatif sur les performances en apprentissage parcimonieux (AP), en introduisant des biais ou des corrélations inutiles.

Les modèles fondation actuels sont souvent pré-entraînés sur des ensembles de données massifs, sans discrimination explicite quant à la pertinence des exemples pour les tâches en aval. Cette approche soulève une question fondamentale : est-il possible d'améliorer les performances d'un modèle en AP en réduisant et en sélectionnant les données utilisées pour son pré-entraînement ?

Ce chapitre propose une approche centrée sur les données, visant à identifier et à ignorer les sous-ensembles nuisibles ou non pertinents, tout en optimisant l'utilisation des échantillons utiles pour les tâches cibles. Étant donné que chaque pré-entraînement est coûteux en calcul, nous choisissons de nous concentrer sur l'ajustement fin (*fine-tuning*) d'un modèle généraliste sur un sous-ensemble des données ayant servi à son pré-entraînement. Ainsi, nous tentons d'"oublier" certaines classes qui pourraient être néfastes pour la performance d'une tâche *few-shot*.

Identification des sous-ensembles pertinents

Pour déterminer quel sous-ensemble de classes de pré-entraînement est pertinent, nous considérons trois scénarios :

— **Informé par la Tâche (IT)** : Dans ce cas, nous avons accès aux données de la tâche (l'ensemble support). Nous sélectionnons les classes de pré-entraînement les plus similaires aux données de la tâche cible. Pour ce faire, nous identifions les K classes les plus activées par le classifieur utilisé lors du pré-entraînement, lorsqu'il est appliqué à la sortie de l'extracteur de caractéristiques.

— **Informé par le Domaine (ID)** : Dans ce cas, nous avons accès à des images du domaine cible. Cela correspond, par exemple, à une situation où un robot prendrait des photos non annotées issues de la même distribution que celle de la tâche. Nous sélectionnons les classes de la même manière qu'en IT, mais en utilisant des données non annotées et abondantes. Cette approche offre un ciblage moins variable, mais plus biaisé, car le domaine est plus large que la tâche d'intérêt sur laquelle l'extracteur de caractéristiques sera évalué.

— **Non-informé (NI)** : Dans ce cas, nous considérons qu'il n'est plus possible de

Table 2 – Changement de performance Δ lorsqu'on applique la méthode proposé (ajustement fin sur un sous-ensemble de classes) en utilisant la méthode informé par la tâche (IT) ou les données (ID) (CC-BY).

Jeu de Données	Méthode	1-shot 5-ways		5-shots 5-ways		MD	
		Référence	Δ	Référence	Δ	Référence	Δ
Aircraft	IT	39.95 ±0.70	-0.06 ±0.33	63.18 ±0.74	**+0.26 ±0.31**	65.86 ±0.90	+1.33 ±0.25
	ID		**+0.34 ±0.32**		**+0.54 ±0.31**		+1.32 ±0.27
CUB	IT	64.34 ±0.90	+2.64 ±0.44	87.78 ±0.59	**+2.16 ±0.26**	79.29 ±0.90	+1.08 ±0.19
	ID		**+3.27 ±0.44**		**+2.29 ±0.26**		**+2.20 ±0.20**
Moyenne sur 9 jeu de données	IT		**+1.43 ±0.38**		**+1.39 ±0.28**		**+1.31 ±0.21**
	ID		**+1.73 ±0.57**		**+1.55 ±0.41**		**+1.61 ±0.30**

procéder à un ajustement fin (comme en IT) lorsque la tâche est révélée, et qu'il n'existe pas de données non annotées (comme en ID). Nous partitionnons alors les classes en utilisant un regroupement (*clustering*) basé sur la méthode de Ward, appliquée à des représentations visuelles et/ou sémantiques (noms des classes). En appliquant certaines heuristiques, nous sélectionnons *a posteriori* l'un des modèles ajustés (*fine-tuned*) sur chacun des groupes de classes.

Résultats et Interprétations

Nous observons une augmentation significative des performances (y compris en utilisant des intervalles de confiance ouverts) lorsque nous ajustons sur un sous-ensemble de classes sélectionnées avec information (IT ou ID). Comme indiqué dans la table 2, cela n'est cependant pas toujours le cas. Sur certains jeux de données, on observe des baisses de performances. Comment expliquer cela ? Les jeux de données où la méthode fonctionne sont ceux où des classes similaires sont présentes dans le jeu de données de pré-entraînement (ImageNet). C'est le cas pour CUB, un dataset d'oiseaux. En revanche, Aircraft ne possède qu'une seule classe correspondante dans ImageNet (Airliner), ce qui entraîne de fortes pertes de performances. En moyennant sur tous les jeux de données, nous observons un gain de performance significatif, et ce, dans chaque type d'échantillonnage de tâche (1-shot 5-ways, 5-shots 5-ways et échantillonnage Metadataset [8]). On note également que les effets sont plus marqués (à la hausse comme à la baisse) lorsque l'on est informé par le domaine, ce qui suggère que les tâches sont trop biaisées pour permettre une sélection optimale des données adaptées.

En Non-Informé (NI), nous observons également des gains robustes, mais globalement inférieurs à ceux obtenus avec information. Nous testons un grand nombre d'heuristiques

pour la sélection d'extracteur de caractéristique. En 5-shots 5-ways, MCS se distingue comme la meilleure heuristique. Nous constatons que le groupement des classes n'a pas d'effet bénéfique significatif, car l'application des heuristiques sur une librairie d'extracteurs de caractéristiques ajustés sur des sous-ensembles aléatoires fonctionne presque aussi bien qu'avec des sous-ensembles choisis par groupement (en utilisant des caractéristiques visuelles et/ou sémantique).

Conclusion

La thèse s'est concentrée sur l'amélioration de l'Apprentissage Parcimonieux (AP) en adoptant une approche centrée sur les données. L'un des principaux axes explorés a été le développement de méthodes pour entraîner des extracteurs de caractéristiques robustes. En combinant des techniques comme l'augmentation de données, l'utilisation d'ensembles de modèles et l'apprentissage auto-supervisé, la thèse a montré qu'il était possible d'atteindre des performances élevées sur des benchmarks standards, tout en simplifiant le processus d'entraînement.

Un autre aspect clé a été la révision des méthodes d'évaluation des performances en AP. La distinction entre intervalles de confiance fermés et ouverts a permis de révéler des biais dans les conclusions des benchmarks actuels, en mettant en évidence l'importance de mieux prendre en compte la variabilité des données. Des tests par paire et le dimensionnement de tâche ont également été proposés pour renforcer la robustesse des comparaisons entre modèles.

Enfin, la thèse a démontré que toutes les données de pré-entraînement ne contribuent pas de manière équivalente à la généralisation. Certaines classes peuvent introduire des biais ou des corrélations inutiles. Une approche consistant à sélectionner les classes les plus pertinentes ou à ajuster les modèles sur des sous-ensembles spécifiques a permis d'améliorer les performances sur les tâches cibles.

Cette thèse apporte des avancées significatives dans le domaine de l'Apprentissage Parcimonieux (AP). Elle remet en question l'idée très répandue selon laquelle le pré-entraînement réalisé en une seule fois sur le plus vaste ensemble de données possible constitue une solution universelle pour généraliser efficacement aux tâches d'AP. Des approches plus progressives avec des modèles spécialistes sont de sérieux compétiteurs.

Table of Contents

1 Introduction **21**

Introduction **21**

1.1 Historical context of DL . . . 21
1.2 Data and compute are the cornerstone of AI . . . 22
1.3 Open Problems in AI . . . 24
1.4 Few-Shot Learning . . . 25
1.5 Problem statement . . . 27
1.6 Contributions and Outline . . . 27
1.7 Toolbox of Deep Learning . . . 28
1.7.1 Architectures . . . 29
1.7.2 Training . . . 35
1.8 Usual methods in Few-shot Learning . . . 37
1.8.1 Few-Shot Learning 101 . . . 37
1.8.2 Why do we need a pre-training? . . . 38
1.8.3 Adaptation to the few-shot task . . . 39
1.9 Notations and Glossary . . . 40

2 Training A Robust Feature Extractor **45**

2.1 Introduction . . . 46
2.2 Related Work . . . 48
2.2.1 Data augmentation . . . 49
2.2.2 Backbone training . . . 49
2.2.3 Exploiting multiple backbones . . . 50
2.2.4 Few-shot classification . . . 50
2.3 Methodology . . . 51
2.3.1 Backbone training (Y) . . . 52
2.3.2 Augmented samples (AS) . . . 53
2.3.3 Ensemble of backbones (E) . . . 53

2.3.4 Feature vectors preprocessing . . . 53
2.3.5 Classification . . . 54
2.4 Results . . . 55
2.4.1 Ranking on standard benchmarks . . . 55
2.4.2 Ablation study . . . 62
2.4.3 Discussion . . . 63
2.5 Transductive tests with imbalanced settings . . . 63
2.6 Conclusion . . . 67
2.7 What would we do differently now? . . . 67

3 Assessing Accuracy Uncertainty while considering task inter-depedance 69
3.1 Introduction . . . 69
3.2 Closed CIs vs. Open CIs . . . 72
3.2.1 A mathematical description of the problem . . . 72
3.2.2 Are OCIs larger than CCIs? An empirical study . . . 75
3.2.3 Impact on Conclusiveness . . . 77
3.3 Paired tests . . . 77
3.3.1 Definitions . . . 77
3.4 Sizing tasks to narrow OCIs . . . 80
3.4.1 Mathematical derivation of $\mathrm{Var}(\bar{A})$ . . . 81
3.4.2 Effect of S and N on Q^* . . . 83
3.5 Benchmark Proposal . . . 87
3.6 Related Work . . . 90
3.7 Limitations . . . 91
3.8 Conclusion . . . 92
3.9 What would we do differently now? . . . 92

4 Less is More: How focusing a model on less data can improve downstream Few-Shot Accuracy? 95
4.1 Introduction . . . 96
4.2 Background and related work . . . 97
4.3 Feature extractors for fewer base classes . . . 100
4.3.1 Formulation . . . 100
4.3.2 Choosing class subsets: Informed settings . . . 102
4.3.3 Choosing class subsets: Uninformed setting . . . 103

4.3.4 Heuristics for selecting a feature extractor . . . 104
4.4 Experiments . . . 107
4.4.1 Effect of informed class selection . . . 107
4.4.2 Uninformed setting . . . 112
4.4.3 Implementation details . . . 114
4.4.4 Discussion . . . 115
4.4.5 Ablation study on the number of selected classes . . . 115
4.5 Conclusion . . . 117
4.6 Appendix . . . 118
4.6.1 Impact of learning rate on fine-tuning (DI selection) . . . 118
4.6.2 About batch normalization during fine-tuning . . . 118
4.6.3 A closer look at the unsupervised selection of classes . . . 118
4.6.4 Logistic Regression Few-Shot Classifier . . . 119
4.6.5 Training from Scratch . . . 120
4.6.6 Silhouette scores . . . 120
4.6.7 Segmentation Tasks . . . 122
4.6.8 Feature space distortion or better features? . . . 122
4.6.9 Support set fine-tuning . . . 124
4.7 What would we do differently now? . . . 124

5 General Conclusion **127**
5.1 Summary of contributions . . . 127
5.2 General answer to the Problem statement . . . 128
5.3 Outlooks and Future works . . . 128
5.3.1 What is a real-world scenario? . . . 128
5.3.2 Feature Space Dynamics in transfer learning: Onthological relations bewteen source and target . . . 130
5.3.3 Alternative Methods to Incorporate Semantic Information in the FSL-VLM Framework . . . 131
5.3.4 The Future Role of LLMs in FSL . . . 132

Bibliography **133**

References **133**

Appendix **151**
5.4 Appendix of Chapter 2 151
5.4.1 Influence of the temperature in the transductive setting 151
5.4.2 Influence of the number of crops 151
5.4.3 Influence of the number of backbones 152
5.5 Appendix of Chapter 3: Additional results on the benchmark 154
5.6 Appendix of Chapter 4 157
5.6.1 Other tables and Figures 157

Chapter 1

Introduction

Contents

1.1 Historical context of DL 21
1.2 Data and compute are the cornerstone of AI 22
1.3 Open Problems in AI 24
1.4 Few-Shot Learning 25
1.5 Problem statement 27
1.6 Contributions and Outline 27
1.7 Toolbox of Deep Learning 28
1.7.1 Architectures 29
1.7.2 Training 35
1.8 Usual methods in Few-shot Learning 37
1.8.1 Few-Shot Learning 101 37
1.8.2 Why do we need a pre-training? 38
1.8.3 Adaptation to the few-shot task 39
1.9 Notations and Glossary 40

1.1 Historical context of DL

For the past 15 years, the field of Machine Leaning (ML) made spectacular progress. From its infancy with hand-written digits recognition [1] to its current developments with video generation from text prompts [11] and protein folding prediction [2], Deep Learning (DL), a sub-field of ML, is becoming increasingly prevalent in science [2, 12], culture [13], industry [14], defense [15] and the general zeitgeist of technology [16]. As such, 2020 marked the advent of generative Artificial Intelligence (AI) available for regular consumer in particular with the release of tools such as ChatGPT and Mid-Journey. One may wonder as to why such tremendous progress was made in a short time frame.

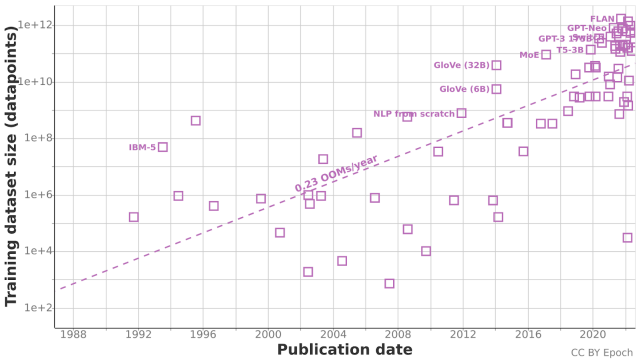


Figure 1.1 – Source from https://epochai.org/blog/trends-in-training-dataset-sizes (CC-BY)

1.2 Data and compute are the cornerstone of AI

Data and Compute These remarkable achievements were made possible thanks to the increased availability of two fundamental ingredients: data and compute. First, data is now extremely abundant. In 2023, it has been estimated that 120 zetta-bytes of information were created [17]. The rapid expansion of cheap and ubiquitous sensors such as smartphone cameras and the internet improved data quantity and accessibility, thus making training datasets in AI reach unprecedented sizes as illustrated on Figure 1.1.

This increase in dataset size being processed can also be explained by the increase in computing power which permits a higher bandwidth of data processing [18]. Computing power for AI was largely improved thanks to the parallel computing design of Graphics Processing Units (GPUs) originally created to accelerate computer graphics and image processing [18, 19, 20]. This parallel architecture was perfectly fitting the highly parallelized structure of DL models. According to the Figure 1.2, state-of-the-art machine learning systems today require 10^{10} times more training compute than at the start of the deep learning era reflecting the evolution of the hardware [21].

Scaling laws This relation between AI advances and the growth in data and compute has been studied more formally under the framework of the so-called "neural scaling laws" [22, 23]. Scaling laws exhibit statistical links between, on the one hand, the number of trainable parameters, the size of the dataset (D) or the computational cost measured in FLOPs and, on the other hand, the performance in the model measured with its loss (L) or accuracy. [23] found that $L = \beta D^{-\alpha}$ with β as a function of the model architecture and other parameters while α remains only sensitive to the task of choice (such as language,

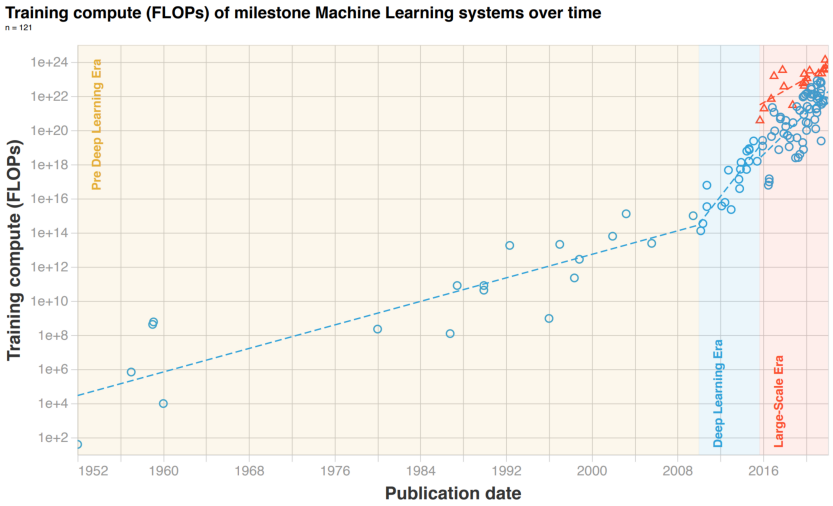


Figure 1.2 – © [2022] IEEE. Reprinted, with permission, from Jaime Sevilla; Lennart Heim; Anson Ho; Tamay Besiroglu; Marius Hobbhahn and Pablo Villalobos , Compute Trends Across Three Eras of Machine Learning, 2022 International Joint Conference on Neural Networks (IJCNN)

vision or audio).

Consequences in terms of power consumption This dependence on compute has a large impact on the environment, the evolution of the research community, and economic consequences of AI [24, 25, 26]. As estimations of carbon emission are often disputable since many factors are at play such as the source of energy, we prefer not to give world-scale numbers here. Nonetheless, in the following, we choose to give a small-scale estimation of power-consumption for a certain task. Please, note that such comparison is a simple highlight and certainly does not accurately represent the full consumption of AI models across the globe and even less its environmental impact. The generation of 1000 images has been estimated to cost approximately 2.9 kWh of energy [27]. The reader may thus compare it three standard microwaves ovens running at full power (1000W) for an hour.

This reliance on data and compute ushered the current "Large-Scale Era" as described in Figure 1.2. As its name suggests, this era is characterized by the development and utilization of large-scale foundation models. They are highly expressive *universal* models. These models are particularly hard to train due to the data and compute requirements needed to train them [28]. This splat the research community and economic interests into two groups: those who can train large foundation models, typically large tech companies, and those who can only use them [24]. This thesis falls into the second category. It aims

at developing methods in the challenging cases where data is particularly scarce. Next, we focus on the open-problems remaining in AI research.

1.3 Open Problems in AI

Among the many, open scientific problems in AI we rapidly delve into three of them: explanability, privacy and generalization. The last one playing a central part in this thesis.

XAI or eXplainable Artificial intelligence is one of those problems. This problem stems from the fact that in many cases a black-box model cannot be integrated in real-world processes since trusting it would require a minimum amount of understanding of its decision processes. More precisely, let us first define *explainability*:

> *"**Explainability** [...] aims for making (a) the context of an AI system's reasoning, (b) the model, or (c) the evidence for a decision output accessible, such that they can be understood by a human"* [29, 30].

This may lead to the *interpretability* which refers to the model's decision being explainable and the purpose understood [31]. XAI is mostly demanded for explaining visual systems or language models with applications in medical imaging and recommendation systems [30]. For instance, in medical imaging, attention or saliency maps on a CT (Computed Tomography) scan can inform a physician on where a problem might have been detected [30]. In an idealized way, such information might in turn lead the physician to dismiss a model's prediction given that a potential confounder, not taken into account by the model, is identified by the human physician. A deeper understanding of the decisions and failure cases is of utmost importance for many industries [32]. In the next paragraph, we focus on another major open problem in AI.

Privacy Our second open-problem is about privacy. If AI models become ubiquitous and run on servers, this means that sensitive or private *unencrypted* data might have to be remotely stored on potentially adverse or unsafe servers. Such problem is addressed by processing information on encrypted data which is what homomorphic encryption does [33, 34] at the cost of a massive computational overhead. Another approach is to decentralize the training though federated learning, thus avoiding centralization of information, compute and data storage [35, 36]. Furthermore, the usage of a model can reveal information about the content of its training dataset. Membership Inference Attack aims

at predicting if a sample was in the training set of a model [37]. Some method can even reconstruct samples from the dataset [38]. Another direction, differential privacy [39] also focuses on datasets. Differential privacy provides mathematical guarantees that a dataset, while remaining relevant for machine learning training, reveals no information (or only a measurably small amount) about individual samples. This ensures that the privacy of individuals in the dataset is protected, even when the data is used for analysis or training models. Next, we focus on the open problem that is the most related to this thesis.

Generalization Finally, generalization plays a key part of this thesis and has been a major problem in AI since its inception [40]. It is at the core of learning itself. Generalization, as opposed to memorization, extends and abstracts the given limited information to successfully apply it to a broad set of unseen problems. The ability to generalize is key to the real-world usage of AI solutions. To address this issue many works have focused on domain adaptation [41] and robustness which is the ability to retain accuracy despite variation in the input data [42]. This thesis focuses on generalization from samples and thus on task and domain adaptation in the context of few-shot learning. First, we will delve into what few-shot learning is and see why pre-training is needed. We then dig into different standard approaches to pre-train and adapt models.

1.4 Few-Shot Learning

Few Shot Learning Few-Shot Learning (FSL) emerged as one of the major challenges of AI. The question FSL aims to answer is how to generalize knowledge from few examples. This question arises from the remarkable ability of humans to learn new concepts from very few examples, whereas computer vision systems appear to be particularly limited in this task [43]. Consider the mushrooms "Pleurocybela porrigens" and "Tolypocladium capitatum." Most people are unfamiliar with these species by their scientific names. However, if introduced as "Angel wings" and "Round-headed truffle club," or through a quick online search that shows a few images of each, assuming there are only two possible choices, the anyone could soon learn to distinguish between them during a forest walk. This task, though straightforward for humans, poses a considerable challenge for AI models not specifically trained on these categories. FSL aims to equip models with the capability to learn and make accurate predictions from minimal data. The primary challenge lies in the fact that each training sample represents a specific instance, complete with its own

biases and atypical features. Moreover, the characteristics of interest, such as the object, background, and color, are not always clearly defined. What is more, FSL extends beyond image recognition, as it is also widely applied in various other domains such as sound, language, and EEG (ElectroEncephaloGraphy) analysis and other tasks such as segmentation or object detection. In the past few years, Few-Shot Learning became particularly popular with applications in defence [44] and health [45].

In this thesis, we focus on image classification for two reasons, (1) this domain is usually one of the forefront of AI research [1] and has excellent benchmarks (2) my team at Institut Mines Telecom IMT Atlantique and Australian Institute for Machine Learning AIML are particularly specialized in computer vision.

Current approaches to Few-Shot Learning Most methods today in FSL rely on the adaptation of large foundation models. As such, the main goal of FSL consists in adapting these universally capable models to specific tasks, using only few examples. This is a particularly interesting change of paradigm with classical machine learning, where the goal is no longer to generalize but instead to specialize. Obviously, the main difficulty here is to find the good way to specialize, circumventing the unavoidable biases that come from the very limited pool of training examples.

From the outset of this thesis, we were intrigued by a fundamental question: if a foundation model is capable of performing a wide range of tasks (general purpose), how does it discern which aspects of the examples given for few-shot tasks are relevant? For illustration, a foundation model would typically focus on colors in input images, while these colors could be orthogonal to the task at hand. Is it possible to select only the portion of the foundation model that is meaningful for our task based on the given few examples?

Answering this scientific question requires going through multiple steps: 1. building efficient feature extractors from large datasets, 2. finding reliable ways to measure performance on few-shot tasks and 3. proposing methods to efficiently select the features of the model that are useful for the considered few-shot task.

1. Language has become another forefront recently.

1.5 Problem statement

The starting point of this thesis is the following observation: certain training data can be detrimental or misleading for specific tasks. Consider for example the two problems of learning to differentiate between triangles and squares, and learning to differentiate between black and white shapes. These two problems can be easily solved using foundation vision models, but each requires distinct features. Real-world problems can be more intricate. Therefore, a better way to frame the issue is as follows: when the feature extractor is trained, it is encouraged to focus on specific features relevant to the pretext task it was trained on. However, when deployed for a specific few-shot task, only some of these features may be relevant, while others might have an adverse effect by highlighting unimportant or confusing elements in the scene.

The main goal of this thesis is to investigate the existence of adversarial samples in the training set of the feature extractor.

Identifying adversarial samples and developing strategies to mitigate their effects can significantly enhance our understanding of what a model is learning, which in turn can advance Explainable AI (XAI). Additionally, focusing on few-shot learning, where a model can adapt to a new task on the fly, holds great promise for ensuring privacy, as there is no need to upload or share data. Finally, this question has strong ties with the theory of generalization, understood as from a pretext task to downstream tasks. As such, the positioning of the thesis is at the core of central and important questions in the field of Machine Learning today.

1.6 Contributions and Outline

In the next paragraphs, we briefly outline the main contributions of this thesis. Each of the three contributions are form a chapter in the thesis.

— **Chapter 2: Making Robust Models** Firstly, we study the optimal way to make it an effective *robust* feature extractor for FSL. By incorporating multiple standard techniques, such as data augmentation and the inclusion of a self-supervised loss during pre-training, we demonstrate that state-of-the-art performance can be achieved for in-domain classification.

Highlight: We could obtain state-of-the-art performance on certain benchmarks showing using a set of tools improving the robustness of our models.

— **Chapter 3 Accurately assessing the performance of FSL:** In this section, we focus on establishing a fair and reliable evaluation of Few-Shot Learning (FSL) methods. A method is deemed superior if it consistently achieves higher accuracy compared to others. We emphasize the importance of computing confidence intervals in FSL evaluations. Our analysis reveals that the predominant method fails to account for data randomness and is therefore specific to particular datasets. We propose methods to enhance the robustness of conclusions in FSL comparative studies while incorporating the randomness of the data.

Highlight: We demonstrate that claims of one method outperforming another are often only valid within the specific context of a benchmark. When data randomness is considered, these claims may be inconclusive. Alarmingly, we discovered an instance where our assessment methods reversed the original conclusion.

— **Chapter 4: Adapt to task using abundant and relevant data** Finally, we propose particularization through a data-centric method. This method is rare in the sense that it neither alters the architecture nor modifies the training process to achieve particularization. We demonstrate that certain segments of the pre-training dataset can adversely affect generalization on specific tasks. We illustrate that "forgetting by fine-tuning" specific portions of the pre-training dataset can significantly enhance performance on cross-domain Few-Shot Learning (FSL) tasks. This approach reallocates the feature space to better suit the task or domain of interest.

Highlight: We show that significant improvements can be obtained by "better using" the pre-training dataset. Sub-domains of the pre-training domain are either better suited or detrimental to the transfer to downstream tasks.

1.7 Toolbox of Deep Learning

In this section, we delve into the key components that constitute Deep Learning today. We will start by discussing the architectures used and conclude with the methods for training these untrained models.

1.7.1 Architectures

Unlike many significant results in classical Machine Learning (ML), Deep Learning (DL) differentiates itself in several notable ways: (a) relevant features emerge from end-to-end training, meaning that important representations of the input signal are not hand-crafted or manually selected by humans, and (b) architectures are *deep*, consisting of stacked layers made up of various modules. DL practitioners focus on designing architectures using inductive biases based on their understanding of the data to frame the appropriate modules. For instance, computer vision tasks involving many 2D images have led to the widespread use of Conv2D modules, as discussed below.

Modules

Linear layers is the simplest module. It consists in a simple matrix multiplication with a bias term.

$$\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{X} + \boldsymbol{B} \tag{1.1}$$

where $\boldsymbol{Y} \in \mathbb{R}^{d_Y}, \boldsymbol{X} \in \mathbb{R}^{d_X}, \boldsymbol{W} \in \mathbb{R}^{d_Y \times d_X}$ and $\boldsymbol{B} \in \mathbb{R}^{d_Y}$, respectively represent the output vector, the input vector, the matrix of weights and the bias vector. The coefficients in $\boldsymbol{W}$ and $\boldsymbol{B}$ are parameters that can be learned during training.

Convolutions [46] used to be the cornerstone of computer vision until recently. Given a 2D image, one can make the following observation: local structures, such as edges, are critical to understand what the image consists in. In fact, two observations lead to this observation. (a) If pixels were randomly shuffled the image would be unintelligible even by a human. (b) An image shifted by a simple translation does not change. Consequently, using an equi-variant local region pattern detector is likely create a relevant representation of the image. This is what convolution modules consist in. Mathematically a convolution filter acts according to Equation 1.2.

$$Y(i,j) = (X * K)(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X(i+m, j+n) \cdot K(m,n) \tag{1.2}$$

where $\boldsymbol{Y} \in \mathbb{R}^{(H-M+1)\times(W-N+1)}, \boldsymbol{X} \in \mathbb{R}^{H \times W}, \boldsymbol{K} \in \mathbb{R}^{M \times N}$ respectively represent the output 2D tensor, the input 2D tensor and the kernel. $\cdot$ is the element-wise product. While this equation is a simple instantiation of the convolution operation, there are many parameters

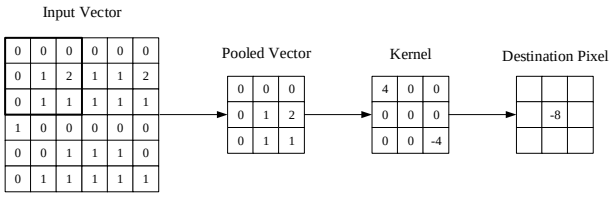


Figure 1.3 – Illustration of the convolution with a 3x3 kernel. (CC-BY) [46]

such as the size of the kernel M and N, the *stride* and *padding* that can influence the design of the operation [47]. Here the coefficients in $\boldsymbol{K}$ are trainable parameters. Figure 1.3 illustrates this operation.

Attention Modules have been transformative since their inception [48]. They first played a role as part of *Transformers* (a larger module comprising attention modules) [48] in language tasks such as translation. Attention modules are designed to dynamically focus on the most relevant parts of the input data. They can thus consider a subset of the signal as part of the context given by the rest of the signal. With a few years delay, it was adopted by the vision community of DL [3] and proved to be an excellent tool for foundation models.

How are images fed to attention modules? Contrary to convolutions where the 2D structure of the image is necessary to process the image, attention modules require something akin to a sequence of tokens as practised in language models[48]. In *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*[3], the authors propose to divide images into square patches of $c \times c$ pixels. By unraveling, one can transform an image of size $\hat{\boldsymbol{X}} \in \mathbb{R}^{H \times W}$ into the representation $X \in \mathbb{R}^{L \times c^2}$ with $L = \frac{HW}{c^2}$.[2]

Self-Attention explained In the following, we mathematically discuss the self-attention module. It is the most widely used version of the attention module in vision. It consists in an interaction between three different embedding of the input signal $\boldsymbol{X} \in \mathbb{R}^{L \times d_X}$ with L the sequence length and d_X the dimension of X.

2. This can be generalized to RGB (Red Green Blue) images

$$\boldsymbol{K} = \boldsymbol{W_K X} \tag{1.3}$$
$$\boldsymbol{Q} = \boldsymbol{W_Q X} \tag{1.4}$$
$$\boldsymbol{V} = \boldsymbol{W_V X} \tag{1.5}$$

The learnable parameters are the coefficients in $\boldsymbol{W_K}, \boldsymbol{W_Q}, \boldsymbol{W_V} \in \mathbb{R}^{\boldsymbol{d \times d_X}}$. They interact though the attention module:

$$\boldsymbol{Y} = \text{softmax}\left(\frac{\boldsymbol{QK^T}}{\sqrt{d}}\right)\boldsymbol{V} \tag{1.6}$$

$\boldsymbol{QK^T}$ is the matrix of similarity between keys and queries. Dividing by $\sqrt{d}$ (for stability) and applying the softmax function yields the attention weights $\boldsymbol{\alpha} \in \mathbb{R}^{L \times L}$. Softmax, often employed in the final layer of classification networks, converts logits into probabilities, ensuring that the output vector sums to one. These values are then used to weight the sum of values in $\boldsymbol{V} \in \mathbb{R}^{L \times d}$. W_K and W_Q are then trained to identify and match different patterns, thereby learning which elements should be linked and how they should influence the output embedding.

Non-linarities or Activation functions Let us consider the MLP [49] or (Multi-Layer Perceptron). It consists in several stacked linear layers. The linear layers must be separated by non-linear Activation functions otherwise the operation could be done in a single linear operation $\boldsymbol{V}$.

$$\boldsymbol{Y} = \boldsymbol{W_0 W_1 W_2 \dots W_n X} = \boldsymbol{V X} \tag{1.7}$$

with $\boldsymbol{V} \in \mathbb{R}^{d_X \times d_Y}$, $\boldsymbol{W_0} \in \mathbb{R}^{m_1 \times d_Y}$, $\boldsymbol{W_n} \in \mathbb{R}^{d_X \times m_n}$, and $\boldsymbol{W_i} \in \mathbb{R}^{m_{i+1} \times m_i}$ for $i \in (1, n-1)$.

Several non-linear activation functions have been used [50], including the sigmoid function ($\sigma(x) = \frac{1}{1+e^{-x}}$), hyperbolic tangent (tanh), ReLU [51], GELU [52] and leaky ReLU [53]. For example, ReLU, or Rectified Linear Unit, is a widely-used activation function defined as relu $= max(0, x)$ providing sparsity and mitigating the vanishing gradient problem by allowing only positive values to pass through.

Other important components MaxPool (Max Pooling), AvgPool (Average Pooling), Dropout, BatchNorm (Batch Normalization), and LayerNorm (Layer Normalization) are

fundamental components in modern DL. MaxPool, used for downsampling, reduces spatial dimensions by selecting the maximum value within non-overlapping subregions, thereby preserving essential features while reducing computation and overfitting. AvgPool is another downsampling operation that reduces spatial dimensions by calculating the average value within non-overlapping subregions, smoothing the feature maps while retaining spatial information. Dropout, introduced around 2012, is a regularization technique that randomly sets a fraction of input units to zero during training, preventing overfitting by encouraging redundant representations. BatchNorm, developed in 2015, normalizes inputs across the mini-batch, addressing internal covariate shift and accelerating convergence by standardizing the mean and variance for each mini-batch. LayerNorm, emerging around the same time, normalizes the inputs across the features within each data sample, stabilizing the learning process and improving generalization by maintaining the mean and variance for each layer's activations. Together, these techniques enhance neural network training efficiency, stability, and performance.

Initialization Some Research has also been performed to study the optimal initialization for a neural net. Indeed, the choice of distribution of weight plays an important role in the performance of the trained model [54]. Another work studies the effect choosing the seed to obtain the best performance [55].

Construction with Modules

Given these parameterized modules, one can simply stack them or have them run in parallel.

In this thesis, we focus on two architectures which come in various flavours and sizes namely the ResNet [56] and the Vision Transformer [3].

ResNets stand for Residual network. This idea came from the idea that deep network can learn more complex patterns [57]. However as models get deeper, the gradient, that is the updating signal flowing backward trough the network (more detail in the next section) either vanishes or explodes. To prevent this, residual blocks, denoted g here, were invented. As shown in Figure 1.4, they consist in a *normal* block with the addition of a skip connection that enables the gradient to flow to deeper parts of the network.

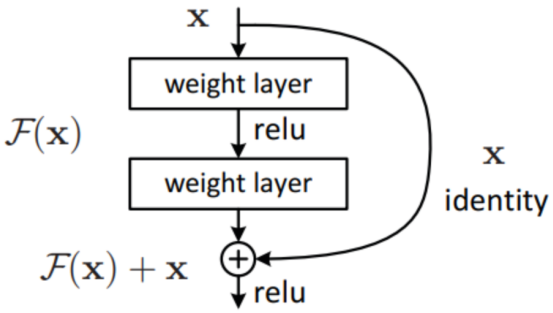


Figure 1.4 – Scheme of the Residual block in ResNets taken from *Deep Residual Learning for Image Recognition*, Kaiming He; Xiangyu Zhang; Shaoqing Ren; Jian Sun, CVPR 2016, IEEE copyright line © 2011 IEEE.

$$g(x) = \overbrace{f(x)}^{\text{normal module}} + \underbrace{x}_{\text{skip connection}} \tag{1.8}$$

ResNets's *normal* module is composed of convolutions and linear layers. Most used denominations are ResNet-10, ResNet-12, ResNet-18, ResNet-50 and ResNet-101. The number after ResNet corresponds to the number of *Residual blocks* with each block typically containing two layers (three for ResNet-50 and Resnet-101).

Vision Transformers Vision Transformers typically use the earlier described transformer with self-attention. The tranformer also contains skip connections, linear layers, layernorm and as depicted in Figure 1.5. Denominations for ViT are typically as follows: ViT-X/c where X can be S, B or L respectively for Small, Base or Large and c is the size $(c \times c)$ of the patch of pixel. A small model has fewer layers and heads than a base or large one. It also has a smaller hidden dimension (latent space dimensionality). Details can be found in Table 1.1.

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Small	8	512	2048	8	22M
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1.1 – Details of Vision Transformer model variants (extended version of Table 1 in [3])

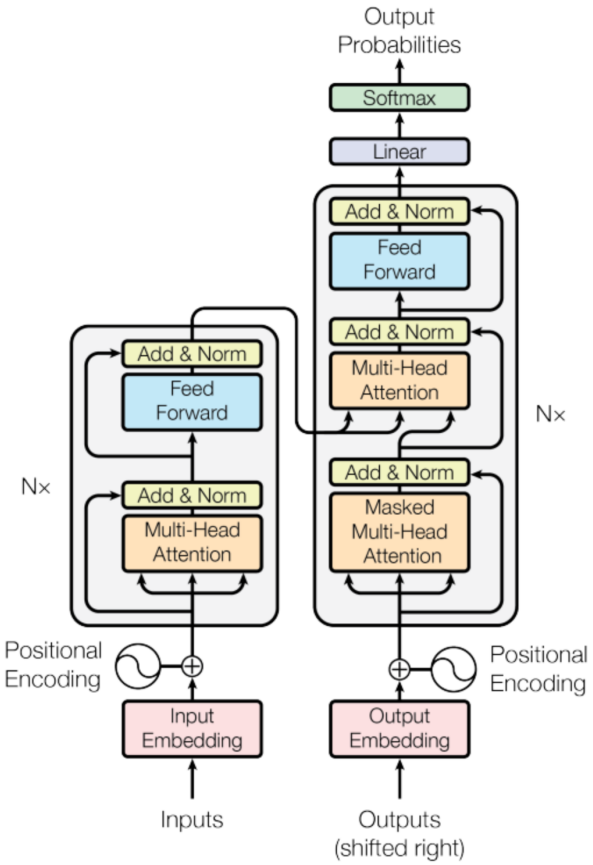


Figure 1: The Transformer - model architecture.

Figure 1.5 – Architecture of a transformer (figure taken from [48]) (Reproduction allowed by Google)

1.7.2 Training

In this section, we explain different methods used to train deep learning models.

A) Supervised Learning In supervised learning, a labeled dataset is available. The model learns to predict the class of each sample accurately. In a classical supervised classification paradigm, the network output, called a *logit*, has a dimensionality of N_c, where N_c is the number of classes in the training dataset. For an input image $\boldsymbol{x}$, the model f will predict a vector $\hat{\boldsymbol{y}} \in \mathbb{R}^{N_c}$. The model progressively improves f such that $\hat{\boldsymbol{y}}$ becomes the one-hot encoding of the true class of $\boldsymbol{x}$ denoted $\boldsymbol{y} \in \mathbb{R}^{N_c}$. One-hot encoding represents a categorical variable with N_c categories as a binary vector $\boldsymbol{e}$ such that $\mathbf{e}_i = \delta_{ij}$ with j the true class of $\boldsymbol{e}$ and δ_{ij} the Kronecker symbol. In this paradigm, the most standard loss/cost/error function is the cross-entropy loss. It is defined as:

$$\mathcal{L}(\{\boldsymbol{y}_i\}_i, \{\hat{\boldsymbol{y}}_i\}_i) = -\sum_{i \in N_s} \boldsymbol{y_i} \log(\hat{\boldsymbol{y}}_i) \tag{1.9}$$

with N_s the number of samples in the dataset.

For this to work effectively, the parameters must be trained though trial and error on all samples of the dataset thanks to an algorithm named *Stochastic Gradient Descent* (SGD) which updates parameters to lower the loss function.

$$\boldsymbol{W^{(t+1)}} = \boldsymbol{W^{(t)}} - \lambda \frac{\partial \mathcal{L}(f_{\boldsymbol{W^{(t)}}}(\boldsymbol{x}), y)}{\partial \boldsymbol{W^{(t)}}}\bigg|_{x,y} \tag{1.10}$$

where $\boldsymbol{W^{(t)}}$ are the parameters at step (trial and error iteration) t, $\mathcal{L}$ is the loss function, $f_{\boldsymbol{W^{(t)}}}$ is the model parameterized by $\boldsymbol{W^{(t)}}$ such that $f_{\boldsymbol{W^{(t)}}}(\boldsymbol{x}) = \hat{y}$ a prediction of the input image $\boldsymbol{x}$ and y is the class label. $\lambda \in \mathbb{R}^+$ is the learning rate, a hyperparameter controlling the update strength.

B) Self-Supervised Learning In the self-supervised learning paradigm, models learn with unlabelled data. How is this possible? Given data one can imagine many sets of tasks that induce the model to discover patterns in the data. For example, one can mask one part of the image and task the model with reconstructing the masked part, or rotate the image and ask the model to predict the applied rotation. Such tasks are not directly classification related yet help produce useful representations for classification and other tasks. In the following, we detail the training of DINO a major self-supervised training

method.

DINO stands for self-**di**stillation with **no** labels. In this method, an input image is *augmented* or transformed twice yielding two versions of the image. Each version goes through either a student model or a teacher model, which typically have the same architecture. The student model learns to mimic the teacher model's output, a process known as distillation. The teacher's parameters are updated not by SGD but thanks to an exponential moving average of the student's parameters, making it self-distillation.

A major challenge is *representational collapse*, where the model outputs trivial solutions to minimize loss. To avoid this, DINO uses centering and sharpening techniques in the teacher model, ensuring features do not collapse. This method produces particularly effective features for k-NN classification. DINOv2, a subsequent version, achieved performance in classification comparable to supervised settings by incorporating better regularization, improved architecture, enhanced datasets, and distillation from large models to smaller models.

Contrastive learning Contrastive learning can be used with or without labels. The idea of contrastive learning is to push the representations of samples from "positive pairs" to be as close as possible from each other while separating or pushing away samples from "negative pairs" . If labels are available, positive pairs correspond to two images of the same class [58]. Without labels, a positive pair is two augmentations (modifications) of the same sample. [59]. Negative pairs, conversely, are simply pairs that are not positive. This straightforward approach yields excellent representations. There are several variants of the contrastive loss which consider different number of pairs, or do not push away negative pairs samples which are sufficiently apart [60].

A particular instance of contrastive learning, also makes use of semantic information. In CLIP [58], pairs of images and their respective captions are processed jointly. CLIP stands for Contrastive Language Image Pretraning. A text encoder creates a text representation of a caption while the corresponding image is encoded creates an visual representation. These matching representations (positive pairs) are then pushed to be close to one another while separating samples from all other pairs.

1.8 Usual methods in Few-shot Learning

In the following, we present typical methods used in Few-shot Learning. We begin by defining an FSL task, which will highlight why FSL frequently involves a pre-training phase followed by an adaptation phase. Subsequently, we will explain the various techniques utilized during the pre-training and adaptation stages.

1.8.1 Few-Shot Learning 101

Few-Shot Training Set or Support Set Shots in FSL are labeled examples. Since our focus is on classification in vision tasks, "examples" should be understood as images. A classification task is said to be *S-shots* when *S labelled samples per class* are available for training. These labels need not contain any semantic information about the class and can simply be integers. For instance, one example might belong to class 0, another to class $i \in [0..K-1]$ with K the *number of classes in the task*. Note that we already made several choices and assumptions: (a) each classes have an equal number of shots and (b) all examples are in classes $[0..K-1]$. More complex versions that do not adhere to these constraints are also used in the literature. All these labelled examples form the support set $\mathcal{S}$. In typical settings $S \in \{1, 5, 10\}$ and $K = 5$.

Few-Shot Test Set or Query Set The query set $\mathcal{Q}$ is the set of unlabelled examples for which we want to predict classes. In benchmarks, the query set is typically balanced to ensure that the accuracy is not biased toward any particular class. Accuracy is calculated as the ratio of correctly classified *queries* to the total number of queries. A typical setting includes $Q = 15$ *queries per class*.

Good representations and ambiguity: the main challenges of FSL Consider a simple 1-shot 2-way task. The shots are a black triangle and white square. This task suffers from both spurious unrelated features and the ambiguity problem [61]. 1) **Feature Representation** Images are simple tensors of shape $(H, W, 3)$ with H, W the height and width and 3 representing the RGB color channels. The challenge is to represent images in a feature space that allows for a simple linear separation of classes. We denote the input image $\boldsymbol{x}$, the feature representation $\boldsymbol{z}$. We obtain that $\boldsymbol{z} = g(\boldsymbol{x})$. g is the feature extractor. It is typically a foundation model without its last classification layer. As mentioned earlier, a foundation feature extractor would have features for both tasks.

2) **Ambiguity in Classification** As humans, we might assume that classifying the breed of the dog is the intended task. However, the model might focus on the background when the task could actually be about the presence of a human, the weather, or even the color of a specific pixel. This ambiguity is central to many open problems in few-shot learning.

1.8.2 Why do we need a pre-training?

Reminder on Deep Learning Modern methods in computer vision are based on deep learning, which relies on modular architectures composed of a large number of parameters. These architectures process images into useful representations or *features* for making predictions, such as classifications.

Parameters are numerous Modern models range from a few millions to several hundred billion parameters. Given the bias-variance tradeoff, this vast number of parameters increases the risk of learning spurious correlations, as over-parameterized models tend to have high variance, which leads to overfitting. Although the double descent phenomenon suggests that increasing parameters can eventually improve generalization, few-shot learning tasks initially face significant overfitting challenges, making effective pre-training essential.

In the previously considered 1-shot 2-way task, starting with a randomly initialized neural network would lead to poor representations because it can easily distinguish the images. For example, simply averaging all pixel values could be enough to train a classifier to distinguish the images. Highly parameterized models would therefore grossly *overfit* such a small training set. Consequently, it is not a viable solution for few-shot learning.

Pre-training as a solution to few-shot overfitting A common methodology is to learn representations using modern models with large amounts of data (excluding the support set). This data forms the *generic* or *base* dataset. Correctly classifying 100+ classes, each containing 1000+ examples, requires the model to process the images such that classes can be linearly separated in the latent space. This large dataset typically reduces ambiguity and allows the model to focus on the intended classes.

Once trained, the pre-trained model provides good representations for classes defined with only a few shots, even if these classes are initially ill-defined. This approach addresses the overfitting issue (2) by leveraging the comprehensive feature space learned from the

base dataset. Additionally, it can partly address the ambiguity problem (1), as the robust representations are likely to be effective for various intended tasks. The final step involves using these representations to classify unseen classes, which is discussed further in this thesis.

1.8.3 Adaptation to the few-shot task

Several methods are employed to adapt to a few-shot task, broadly categorized into two main approaches: "fixed features" and "adapted features".

Fixed Features: In this approach, the model is kept frozen, and only the final classification layer is adjusted to accommodate the few-shot task. This method leverages pre-trained features without altering the underlying model structure. The problem is then reduced to finding the best classification method from feature space to the classes. This can come through either distance-based or though Logistic Regression (LR). For example, a distance-based classifier is the nearest neighbors methods where a sample is classified as the closest shot available using euclidean distance.

One of the most used methods in this thesis is the NCM for Nearest Class Mean where a center of all labeled examples from a class is computed and named *centroid* or *barycenter*.

$$\forall i : \overline{\mathbf{c}_i} = \frac{1}{|\mathcal{S}_i|} \sum_{\mathbf{z} \in \mathcal{S}_i} \mathbf{z} \ , \tag{1.11}$$

then associating to each query the closest centroid:

$$\forall \mathbf{z} \in \mathcal{Q} : C_{ind}(\mathbf{z}, [\overline{\mathbf{c}_1}, \ldots, \overline{\mathbf{c}_n}]) = \arg\min_i \|\mathbf{z} - \overline{\mathbf{c}_i}\|_2 \ . \tag{1.12}$$

Adapted Features: This approach involves transforming either the model itself or its features to better suit the few-shot task. This can include fine-tuning the model or applying adapters such as LORA (LOw Rank Adapter) [62], DORA [63] a derivative of LORA where only certain direction in the weight space are reweighed. In LORA, we consider each matrix of weight representing a linear layer and denote it $W \in \mathbb{R}^{N \times M}$. We train a shift of weights ΔW of low rank: $\Delta W = AB$ with $A \in \mathbb{R}^{N \times R}$ and $B \in \mathbb{R}^{R \times M}$. In this setting the adapted weights are $W' = W + \Delta W$. Instead of training the entire NM parameters, we only train $R(M+N)$ parameters which significantly less since $R << M, N$

Whether using fine-tuning or adapter-based methods, the main challenge is avoiding overfitting. Over-adaptation to the task's support set can significantly hinder the model's ability to generalize to other tasks and, more importantly, even compromise its performance on the specific task it was adapted for. Therefore, selecting the appropriate hyper-parameters is crucial for achieving optimal downstream performance.

In the next chapter, we focus on training highly expressive robust *feature extractors*.

1.9 Notations and Glossary

Table 1.2 – Notation Table

Symbol	Description
α, β, γ	A scalar quantity
$\mathbf{a}$	A vector quantity
$\mathbf{A}$	A matrix
$\mathcal{A}$	A set
$\mathbb{R}$	Set of real numbers
$\mathcal{C}$	Set of classes in a dataset
$\mathcal{D}$	A dataset
K	The number of classes or *ways* in a few-shot task
$\mathcal{K}$	Set of classes in a few-shot task
$\mathbb{N}$	Set of natural numbers
N	The number of samples per class in a dataset
Q	The number of queries
$\mathcal{Q}$	Query Set
S	The number of shots
$\mathcal{S}$	Support Set
T_j	j-th few-Shot task
A_j	Accuracy of measured on task T_j
Δ	Difference of Accuracy
M	Number of classes in the selected subset
c_i	Centroid or barycenter of class i given $\mathcal{S}$
$\mathbf{x}$	Input signal (typically an image)

Continued on next page

Table 1.2 – continued from previous page

Symbol	Description
f	Feature extractor
θ	Parameters
f_θ	*Feature extractor* parameterized by parameters θ
$z = f(x)$	Features of x given the *feature extractor* f
$f_{\mathcal{S}_t}$	Few-shot classifier using the *feature extractor* f and the support set of task $t : \mathcal{S}_t$
h	Ultimate classification layer
$g = h \circ f$	Full model
$\mathbf{y} = h(\mathbf{z}) = g(\mathbf{x})$	Prediction Logits
$\nabla f_\theta(\mathbf{x})$	Gradient of the function f with respect to θ evaluated at $\mathbf{x}$
$\partial f / \partial \mathbf{x}$	Partial derivative of f with respect to $\mathbf{x}$
$\|\mathbf{x}\|$	Norm of $\mathbf{x}$ (L2 if not specified)
$\mathbf{x} \cdot \mathbf{y}$	Dot product of $\mathbf{x}$ and $\mathbf{y}$
$\mathbf{x} \times \mathbf{y}$	Cross product of $\mathbf{x}$ and $\mathbf{y}$
$\mathbf{I}_n$	Identity matrix of size $n \times n$
$\mathbb{E}[X]$	Expected value of the random variable X
$\mathrm{Var}(X)$	Variance of the random variable X
$P(A)$	Probability of event A
$P(A \mid B)$	Conditional probability of A given B
λ_i	i-th eigenvalue of a matrix
$\mathbf{u}_i$	i-th eigenvector of a matrix
σ	Standard deviation
μ	Mean
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2
$\det(\mathbf{A})$	Determinant of the matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	Inverse of the matrix $\mathbf{A}$
$\mathrm{tr}(\mathbf{A})$	Trace of the matrix $\mathbf{A}$

Table 1.3 – Glossary Table

Acronym	**Description**
AA	Average Activation
AI	Artificial Intelligence
AS	Augmented Samples
CCI	Closed Confidence Interval (sampling with replacement)
CI	Confidence Intervals
CL	Contrastive Learning
CLIP	A model (Contrastive Language-Image Pre-training)
DL	Deep Learning
DI	Domain Informed
DINO	A class of pre-trained models (self-**di**stillation with **no** labels)
E	Ensemble of Backbones
FIM	Fisher Information Matrix
GPU	Graphics Processing Unit
HPC	High Performance Computing
FSL	Few-Shot Learning
K-Means	An unsupervised clustering method
LOO	Leave-One-Out
MCS	Monte-Carlo Sampling
ML	Machine Learning
MLP	Multi-Layer Perceptron [49]
NCM	Nearest Class Mean (a classification algorithm) [64]
OCI	Open Confidence Interval (sampling tasks without replacement)
Query	two contexts (a) Sample of the Query Set (test set) to be classified in FSL (b) an element in Attention modules
Query Set	Test set of a few-shot task
UI	Uninformed
UOT	Unbalanced Optimal Transport
PT	Paired Test
ReLU	Rectified Linear Unit [51]
ResNet	Residual Network [56]
RKM	Rank-me [65]

Continued on next page

Table 1.3 – continued from previous page

Acronym	Description
Shot	Labeled sample part of the support set of a few-shot task
SNR	Signal-to-noise Ratio
Support Set	Small (by definition of FSL) training set of a few-shot task
SSA	Support Set Accuracy
SSC	Support Set confidence
SSL	Self-Supervized Learning
Task	Any Machine Learning Problem (most of the time refers to a classification few-shot task in this thesis)
TI	Task Informed
Few-Shot Task	Disjoint support and query sets with the same classes
ViT	Vision Transformer [3]
XAI	eXplainable Artificial Intelligence [30]
Ways	Number of classes in a few-shot task
Y	Double loss architecture described in Chapter 1 (shape of Y)

Chapter 2

Training A Robust Feature Extractor

Contents

2.1 Introduction 46

2.2 Related Work 48

2.2.1 Data augmentation 49

2.2.2 Backbone training 49

2.2.3 Exploiting multiple backbones 50

2.2.4 Few-shot classification 50

2.3 Methodology 51

2.3.1 Backbone training (Y) 52

2.3.2 Augmented samples (AS) 53

2.3.3 Ensemble of backbones (E) 53

2.3.4 Feature vectors preprocessing 53

2.3.5 Classification 54

2.4 Results 55

2.4.1 Ranking on standard benchmarks 55

2.4.2 Ablation study 62

2.4.3 Discussion 63

2.5 Transductive tests with imbalanced settings 63

2.6 Conclusion 67

2.7 What would we do differently now? 67

2.1 Introduction

This chapter is largely based on the content of our paper: *EASY: Ensemble Augmented-Shot Y-shaped Learning: State-Of-The-Art Few-Shot Classification with Simple Ingredients* [66] published in the Journal of Imaging in 2022.

The remarkable performances of deep learning (DL) are generally obtained thanks to training on to large databases. Getting the same performance with extremely limited data may seem paradoxical. Yet transfer learning allows us to transfer this knowledge on smaller tasks.

As mentionned in the introduction, the current paradigm of few-shot learning (FSL) relies on foundation models as starting points for classifying new or *novel* classes. However, these models are trained on vast and often undisclosed datasets, making it uncertain whether the novel classes (i.e., the classes from which few-shot tasks are sampled) are genuinely disjoint from the pre-training dataset. Despite this, the exceptional performance of these *off-the-shelf*, highly expressive models on most few-shot tasks has made them the *de facto* starting point for few-shot learning (FSL) practitioners. It is in fact, rather cumbersome to find data that is totally unseen by these models.

In contrast, the traditional few-shot setting involves a benchmark that specifies both the pre-training (or *base*) dataset and the *novel* dataset from which tasks are sampled for evaluation. This design emphasizes the quest for methods able to learn on unseen data and ensured the fair comparisons of methods. This contribution [4] adheres to this traditional paradigm, although it is somewhat outdated in the current context. We now provide a more detailed description of the base and novel datasets.

— A base dataset contains many examples across numerous classes. Due to its large size, it is suitable for efficiently training DL architectures. A validation dataset is often used conjunction with the *base* dataset. In a standard classification setup, the base dataset is used for training, while the validation dataset serves as a proxy to measure generalization performance on unseen data, aiding in hyperparameter optimization. However, in few-shot learning, the validation and base datasets usually comprise distinct classes, enabling the assessment of generalization performance on new classes [67]. Various strategies can be employed to learn effective feature representations from the base dataset, which will be further discussed in

Section 2.2;

— A *novel dataset*, which consists of classes that are distinct from those of the base and validation datasets. It can be just as big as the *base* dataset. Tasks are sampled from it. The labeled samples are often called the *support* set, and the remaining ones the *query* set. In that case, the number of classes K (named *ways*), the number of *shots* per class S and the number of *query* samples per class Q are given by the benchmark. This setting is referred to as K-way-S-shot learning. Reported performances are often averaged over a large number of runs.

To leverage the knowledge previously acquired by models on a base dataset, a common technique is to remove their final classification layer. These models are then referred to as *feature extractor* or *backbone*. They can be employed to convert the support and query datasets into *feature vectors*, that is a representation of these images which incorporates the priors learned in the base dataset. In this study, we exclude the use of additional data, such as other datasets [68], and do not incorporate semantic information [69]. Prior to the classification task, further preprocessing steps may be applied to the samples and/or the associated feature vectors. Another significant approach involves meta-learning [70, 71, 72, 73, 74, 75], as discussed in Section 2.2.

We distinguish two cases which differ in the amount of accessible information:

— In *inductive* few-shot classification, only the support dataset is available to the few-shot classifier, and prediction is performed on each sample of the *query* dataset independently from each other [76];

— In *transductive* few-shot classification, the few-shot classifier has access to both the support and the full query datasets when performing predictions [77]. This extra information of knowing the whole *query* set (or few-shot test set) is of tremendous help. For example, given enough queries, one might discover clusters in the query set which can then be associated with ill-defined few-shot classes thanks to the support set.

Both cases correspond to real-world situations. Generally, the inductive case corresponds to situations where data acquisition is expensive or particularly difficult. This is the case for FMRI data for example where it is difficult to generalize from one patient to another and collect hours of training data on a patient could be harmful [78]. Alternatively, the transductive case corresponds to situations where data labeling is expensive. Such situation can occur when experts must properly label data but the data itself is obtained relatively cheaply, for instance in numerous medical applications [79, 80].

In the past few years, many contributions have introduced methodologies to cope with few-shot problems. They use many ingredients, including distillation [81], contrastive learning [82], episodic training [83], mixup [84], manifold mixup [67, 85] and self-supervision [67]. As a consequence, it is unclear what the positively impactful ingredients are, and whether their performance are robust across different datasets or settings. More problematically, we noticed that many of these contributions start with sub-optimal training procedures or architectures. Admittedly, they provide significant accuracy boosts using their proposed methods, but this improvements is not always robust when tested with better initial models without the proposed ingredients. We show that many of these contributions report baseline performances that can be outperformed with a simpler training pipeline.

In this chapter, we present a very simple method combining ingredients commonly found in the literature and yet achieving highly competitive performance. These results were presented in our paper [4]. This work is designed to help have a clearer view on how to efficiently and simply implement few-shot classification for real-world applications. We aim to define a new baseline with excellent hyperparameters and training routines to compare to and to start with, on which obtaining an accuracy boost will be much more challenging than starting from a poorly trained backbone. Additionally, we show that a simple approach reaches higher accuracy than increasingly complex methods proposed in the recent few-shot literature.

The contribution from our work presented in [4] are the following:

— We introduced a very simple methodology, illustrated in Figure 2.1, for both inductive or transductive few-shot classification.

— We showed the ability of the proposed methodology to reach or even beat state-of-the-art [86, 73] accuracy on several commmon benchmarks of the field.

— All our models, obtained feature vectors and training procedures are freely available online on our github: `https://github.com/ybendou/easy`

— We also proposed a simple demonstration of our method using live video streaming to perform few-shot classification. The code is available at `https://github.com/RafLaf/webcam`.

2.2 Related Work

To begin with, we emphasize on some of the few-shot classification approaches following the classical pipeline. Note that the methodology we proposed [4] uses multiple

ingredients from those presented hereafter.

2.2.1 Data augmentation

First, *data augmentation* or *augmented sampling* are generally used on the base dataset to artificially produce additional samples. Examples include rotations [67], crops [87], jitter, GANs [88, 89], or other techniques [90]. Data augmentation on support and query sets, however, is less frequent. Approaches exploring this direction include [82], where authors propose to select the foreground objects of images by identifying the right crops using a relatively complex mechanism; and [91], where the authors propose to mimic the neighboring base classes distribution to create augmented latent space vectors.

Other forms of data augmentation combine several samples. *Mixup* [84] and *manifold-mixup* [85] are two different forms of linear interpolation of samples and labels. Both can be seen as regularization methods [84, 85]. Mixup creates linear interpolations at the sample level (see Equation 2.1) while manifold mixup focuses on feature vectors as shown in Equation 2.2 with $\lambda \in (0, 1)$. To maintain consistency, the target labels are also interpolated as in Equation 2.3.

$$x_{AS} = \lambda x_1 + (1 - \lambda) x_2 \tag{2.1}$$

$$z_{AS} = \lambda z_1 + (1 - \lambda) z_2 \tag{2.2}$$

$$y_{AS} = \lambda y_1 + (1 - \lambda) y_2 \tag{2.3}$$

2.2.2 Backbone training

Mixup is used with *Self-supervision* (S2) [67] to make *feature extractors* more robust. Most of the time, S2 is implemented as an auxiliary loss. This loss trains the *feature extractor* to recognize which transformation was applied to an image [92].

A common training strategy is *episodic training*. Its philosophy is to having the same train and test conditions. Thus, the training strategy, often based on gradient descent, does not select random batches, but uses custom batches designed as few-shot tasks [70, 93, 94, 83].

Meta-Learning, that is *learning to learn*, is an important line of research in FSL.

This method typically learns a good initialization network or optimizer such that adaptation to new classes can be learned in a few gradient steps [70, 71, 72, 73, 74, 75]. For those acquainted with meta-learning literature, the *meta-train* phase aligns with the *pre-training step* described in the main approach of this chapter and throughout the thesis. In meta-train, traditional training batches are usually replaced by episodes (i.e., FSL tasks), following the principle that training and testing conditions should closely resemble each other. This process is referred to as episodic training. A work leverages this idea to generate augmented tasks in the training of the *feature extractor* [95].

Contrastive learning or CL aims to train a model to learn to maximize similarities between transformed instances of the same image and minimize agreement between transformed instances of different images [96, 97, 98, 82]. There exist a supervized version of it. *Supervised contrastive learning* is a variant of CL which has been recently used in few-shot classification, where similarity is maximized between instances of a class instead of the same image [99, 81].

2.2.3 Exploiting multiple backbones

Distillation was also used in the FSL literature. It aims at transferring knowledge from a teacher model to a student model by training the latter to match the joint probability distribution of the teacher $p(x, \hat{y}_t)$ with $\hat{y}_t$ the prediction of the teacher model [100, 81].

Ensembling is the concatenation of features extracted by different *feature extractors*. It was used to improve performances in FSL classification [95]. It is a simple alternative to distillation. To limit the computationally expensive training of multiple backbones, it was proposed to use of snapshots [101].

2.2.4 Few-shot classification

Classification methods in the inductive setting are based on simple methods such as nearest class mean [64], cosine classifiers [102] and logistic regression [91].

Thanks to the extra information, more methods can be implemented in the transductive setting. For example, clustering algorithms [82], embedding propagation [103] and optimal transport [104] were used successfully to strongly outperform accuracies in the inductive setting.

2.3 Methodology

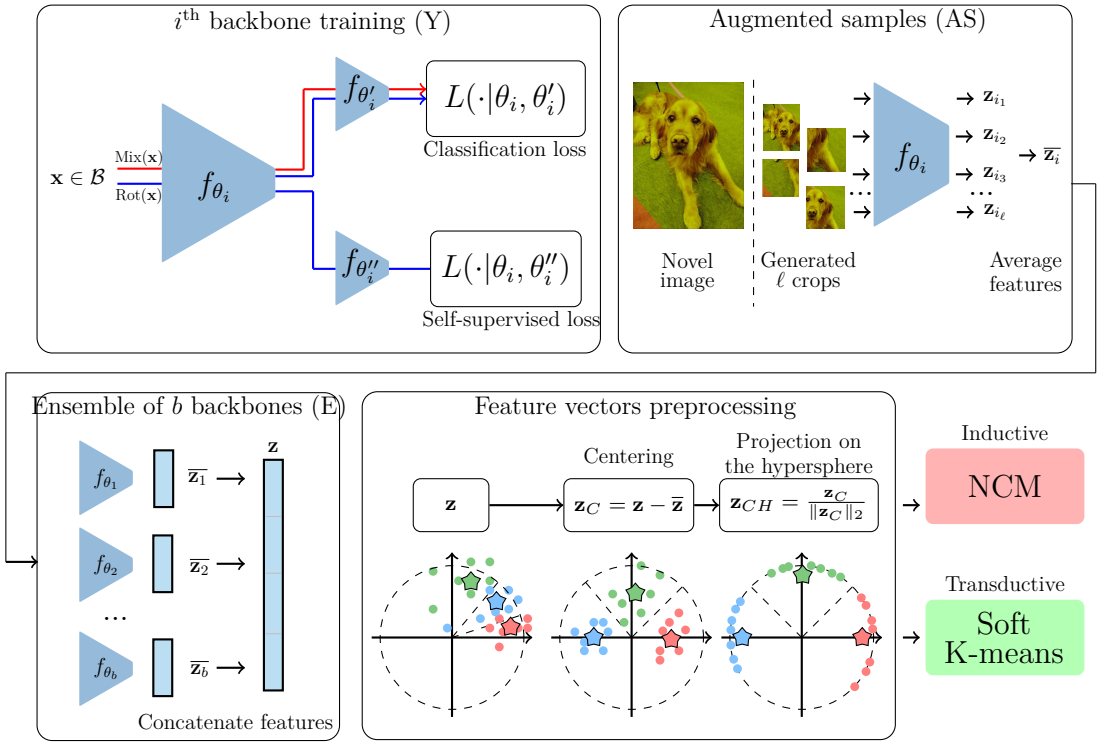


Figure 2.1 – Illustration of the method we proposed [4]. **Y**: First, we first train several *feature extractors* using the base and validation datasets. The base for direct training and validation to select hyper-parameters on FSL tasks sampled from the validation split. We use two cross-entropy losses in parallel: one for the classification of base classes, and the other for the self-supervised targets (rotations). We also make use of manifold mixup [85]. All the *feature extractors* are trained using the exact same routine, except for their initialization seed (for randomness). The sampling order of data batches is also different due to the different seed; **AS**: Then, for each image in the novel dataset and each *feature extractors*, we generate multiple crops, then compute their feature vectors, that we average; **E**: Each image becomes represented as the concatenation of the outputs of AS for each of the trained *feature extractors*; **Preprocessing**: Our pre-processing steps include (a) centering by removing the mean of the feature vectors of the base dataset in the inductive case, or the few-shot run feature vectors for the transductive case, and (b) projecting on the hypersphere. Finally, our classifier is the nearest class mean classifier (NCM) if in the inductive setting or a soft K-means algorithm in the transductive setting. (CC-BY)

The methodology we proposed [4] has 5 steps, described below and illustrated in Figure 2.1. In our experiments, we also report ablation results when omitting the optional steps.

2.3.1 Backbone training (Y)

We augment the data with random resized crops, random color jitters and random horizontal flips. It is standard in the FSL.

We implement a cosine-annealing scheduler [105], updating the learning rate at each step. During a cosine cycle, the learning rate evolves between η_0 and 0. After each cycle, we *warm restart* the learning procedure and start again with a smaller η_0. At the start, $\eta_0 = 0.1$ and reduce η_0 by 10% at each cycle. Our training contains 5 cycles with 100 epochs each.

Our training uses the methodology called S2M2R described in [67]. The idea is to take a standard classification architecture (*e.g.*, ResNet12 [106]), and to plug-in a new logistic regression classifier after the penultimate layer, in addition to the classification head, thus forming a Y-shaped (two heads) model (c.f. Figure 2.1). Our new head is meant to retrieve which one of four possible rotations (quarters of 360° turns) has been applied to the input samples. We use a two-step forward-backward pass at each step, where a first batch of inputs is only fed to the first classifier (for classes), combined with manifold-mixup [67, 85]. A second batch of inputs is then applied arbitrary rotations and fed to both classifiers (classes and rotations). After this training, *feature extractors* are frozen.

Our experiments are performed with a standard ResNet12 as described in [106], where the feature vectors are of dimension 640. These feature vectors are obtained by computing a *global average pooling* over the output of the last convolution layer. Our *feature extractor* contains $\sim$ 12 million trainable parameters. We also experiment with reduced-size ResNet12, denoted ResNet12$\left(\frac{1}{2}\right)$ where the number of feature maps divided by 2, resulting in feature vectors of dimension 320, and ResNet12$\left(\frac{1}{\sqrt{2}}\right)$, (divided by $\sqrt{2}$), resulting in feature vectors of dimension 450. The numbers of parameters are respectively $\sim$ 3 million and $\sim$ 6 million.

As presented in the Introduction, we denote $\mathbf{x}$ an input sample, and f the *feature extractor* then $z = f(\mathbf{x})$ is the feature vector associated with $\mathbf{x}$.

In the following, we use the frozen *feature extractors* to get feature vectors from the base, validation and novel datasets.

2.3.2 Augmented samples (AS)

We generate augmented feature vectors for each sample from the novel dataset. However, We do not augment the validation set because it is particularly computationally costly. We use random resized crops from the corresponding images. We produce multiple versions of each feature vector and average them. The role of augmentations in DL was extensively studied in the literature [107]. It is assumed that most crops contain the object of interest for the classification, thus making the average feature vector relevant.However this does not work for color jitter where it might be a invalid augmentation since some objects rely mainly on colors to be classified (e.g., birds or fruits).

Concretely, we use $\ell = 30$ crops per image since larger values do not improve the accuracy significantly. Here, this step is considered optional.

2.3.3 Ensemble of backbones (E)

To obtain yet greater performance, we proposed to concatenate the feature vectors obtained from multiple backbones (i.e *feature extractor*) trained using the previously described method, but with different random seeds. To ensure fair comparisons, when evaluating a single backbone against an ensemble of b backbones, we adjust the number of parameters per backbone in the ensemble so that the total number of parameters across all b backbones matches that of the single backbone. We consider this strategy to be a viable alternative to distillation, as it avoids the need for additional parameters and offers a relatively straightforward implementation. This step is also optional, and we conduct ablation tests in the next section to evaluate its impact.

2.3.4 Feature vectors preprocessing

Our last two transformations on feature vectors $\mathbf{z}$ are described in [64]. We denote $\overline{\mathbf{z}}$ the average feature vector of the base dataset if in *inductive setting* or of the few-shot considered problem if in *transductive setting*. In the ideal case $\overline{\mathbf{z}}$ would center the vectors of the few-shot tasks around 0 and therefore would be the average vector of the union of the support and query set. However, the number of samples being too small to compute a meaningful average vector in the inductive setting, we instead use the base dataset. In the transductive setting, queries ans shots are used for mean computation. The average vector is therefore less noisy and can be used to compute $\overline{\mathbf{z}}$.

In the first operation (C – centering of $\mathbf{z}$), we compute:

$$\mathbf{z}_C = \mathbf{z} - \overline{\mathbf{z}} \ . \tag{2.4}$$

In the second operation (H – projection of $\mathbf{z}_C$ on the hypersphere), we compute:

$$\mathbf{z}_{CH} = \frac{\mathbf{z}_C}{\|\mathbf{z}_C\|_2} \ . \tag{2.5}$$

2.3.5 Classification

We denote $\mathcal{S}_i$ ($i \in \{1, \ldots, n\}$) the set of feature vectors (preprocessed as $\mathbf{z}_{CH}$) that correspond to the support set of the i-th class, and $\mathcal{Q}$ the set of (also preprocessed) query feature vectors.

In inductive FSL, we use a simple Nearest Class Mean classifier (NCM). First we compute the class barycenters from labeled samples or *shots*:

$$\forall i : \overline{\mathbf{c}_i} = \frac{1}{|\mathcal{S}_i|} \sum_{\mathbf{z} \in \mathcal{S}_i} \mathbf{z} \ , \tag{2.6}$$

then we associate each query to the closest barycenter:

$$\forall \mathbf{z} \in \mathcal{Q} : C_{ind}(\mathbf{z}, [\overline{\mathbf{c}_1}, \ldots, \overline{\mathbf{c}_n}]) = \arg\min_i \|\mathbf{z} - \overline{\mathbf{c}_i}\|_2 \ . \tag{2.7}$$

In transductive FSL, we use a soft K-means algorithm. We compute the sequence below indexed by t. The initial $\overline{\mathbf{c}_i}$ are computed as in Equation (2.6) :

$$\forall i, t : \begin{cases} \overline{\mathbf{c}_i}^0 & = & \overline{\mathbf{c}_i} \ , \\ \overline{\mathbf{c}_i}^{t+1} & = & \displaystyle\sum_{\mathbf{z} \in \mathcal{S}_i \cup \mathcal{Q}} \frac{w(\mathbf{z}, \overline{\mathbf{c}_i}^t)}{\displaystyle\sum_{\mathbf{z}' \in \mathcal{S}_i \cup \mathcal{Q}} w(\mathbf{z}', \overline{\mathbf{c}_i}^t)} \mathbf{z} \ , \end{cases} \tag{2.8}$$

where $w(\mathbf{z}, \overline{\mathbf{c}_i}^t)$ is a weight function on $\mathbf{z}$, that yields the probability of being associated with barycenter $\overline{\mathbf{c}_i}^t$:

$$w(\mathbf{z}, \overline{\mathbf{c}_i}^t) = \begin{cases} \dfrac{\exp\left(-\beta \left\|\mathbf{z} - \overline{\mathbf{c}_i}^t\right\|_2^2\right)}{\displaystyle\sum_{j=1}^{n} \exp\left(-\beta \left\|\mathbf{z} - \overline{\mathbf{c}_j}^t\right\|_2^2\right)} & \text{if} \quad \mathbf{z} \in \mathcal{Q} \ , \\ 1 & \text{if} \quad \mathbf{z} \in \mathcal{S}_i \ . \end{cases} \tag{2.9}$$

Unlike the simple K-means algorithm, we use a weighted average where weight values

are calculated thanks to a decreasing function of the L_2 distance between data points and class barycenters. In this case, we use a softmax adjusted by a temperature value β. Our experiments use $\beta = 5$. This setting led to consistent results across datasets and backbones. Although practically using a finite number of steps, we find the converging centroids $\mathbf{c}_i^{\infty}$. The resulting vectors, predictions are:

$$\forall \mathbf{z} \in \mathcal{Q} : C_{tra}(\mathbf{z}, [\overline{\mathbf{c}_1}^{\infty}, \ldots, \overline{\mathbf{c}_n}^{\infty}]) = \arg\min_i \|\mathbf{z} - \overline{\mathbf{c}_i}^{\infty}\|_2 \,. \tag{2.10}$$

2.4 Results

2.4.1 Ranking on standard benchmarks

First, we compare our method with state of the art on classical settings and datasets. Here are the datasets we use:

— MiniImagenet: A dataset extracted from ImageNet with 64 base classes, 16 validation classes and 20 novel classes. Each class contains 600 images. The resolution is (84x84);

— TieredImagenet: Another subset of ImageNet with 351 base classes, 97 validation classes and 160 novel classes. Classes contain a variable number of samples, usually about 1300. The resolution is (84x84);

— CUB-FS: This dataset (Caltech-UCSD Birds-200-2011) is particularly challenging because it is only composed of pictures of birds. There are a 100 base classes, 50 validation classes and 50 novel classes. The number of images by class is not constant, close to 60. The resolution is (50x50);

— FC-100: It is a subset of CIFAR 100. There are 60 base, 20 validation, 20 novel classes containing 600 images. Images have a low resolution (32x32);

— CIFAR-FS: It is also a subset of CIFAR 100. There are 60 base, 16 validation, 20 novel classes containing 600 images. Images have a low resolution (32x32).

For each FSL method, we detail the number of trainable parameters and the accuracy of 1-shot or 5-shot runs. We use $Q = 15$ query samples per class and results are averaged over 10,000 runs. We present the results in Tables 2.1-2.5 for the inductive setting and Tables 2.6-2.10 for the transductive setting [1]. We report results for the existing methods

1. The codes allowing for the reproduction our experiments are available at https://github.com/ybendou/easy.

using their own reported results in the literature. Some methods do not include their standard deviation over multiple runs and are thus not present in our Tables.

We emphasize that, at the time of its release, our proposed methodology provided new state-of-the-art performance for MiniImageNet (inductive), TieredImageNet (inductive 1-shot setting) and FC100 (transductive), while demonstrating competitive or overlapping results on other benchmarks. Combined with other more elaborate methods, we believe these results could be improved significantly, yielding a new standard of performance for FSL benchmarks. In the transductive case, the proposed methodology [4] is more rarely ranked #1. However, contrary to many alternatives it does not use any prior on the number of samples per class in the generated FSL tasks. We show such experiments in the supplementary material, where we demonstrate that our proposed [4] method greatly outperforms existing techniques when benchmarking on imbalanced classes. Chiefly, our method is simpler than others yet achieves competitive performance over multiple benchmarks.

Table 2.1 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **MiniImageNet** in **inductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	SimpleShot [64]	62.85 ± 0.20	80.02 ± 0.14
	Baseline++ [102]	53.97 ± 0.79	75.90 ± 0.61
	TADAM [108]	58.50 ± 0.30	76.70 ± 0.30
	ProtoNet [83]	60.37 ± 0.83	78.02 ± 0.57
	R2-D2 (+ens) [95]	64.79 ± 0.45	81.08 ± 0.32
	FEAT [109]	66.78	82.05
	CNL [110]	67.96 ± 0.98	83.36 ± 0.51
	MELR [111]	67.40 ± 0.43	83.40 ± 0.28
	Deep EMD v2 [87]	68.77 ± 0.29	84.13 ± 0.53
	PAL [81]	69.37 ± 0.64	84.40 ± 0.44
	invariance-equivariance [112]	67.28 ± 0.80	84.78 ± 0.50
	CSEI [86]	68.94 ± 0.28	85.07 ± 0.50
	COSOC [82]	69.28 ± 0.49	85.16 ± 0.42
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	$\mathbf{70.63 \pm 0.20}$	$\mathbf{86.28 \pm 0.12}$
$36M$	S2M2R [67]	64.93 ± 0.18	83.18 ± 0.11
	LR + DC [91]	68.55 ± 0.55	82.88 ± 0.42
	EASY 3×ResNet12 (ours)	$\mathbf{71.75 \pm 0.19}$	$\mathbf{87.15 \pm 0.12}$

Table 2.2 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **TieredImageNet** in **inductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	SimpleShot [64]	69.09 ± 0.22	84.58 ± 0.16
	ProtoNet [83]	65.65 ± 0.92	83.40 ± 0.65
	FEAT [109]	70.80 ± 0.23	84.79 ± 0.16
	PAL [81]	72.25 ± 0.72	86.95 ± 0.47
	DeepEMD v2 [87]	74.29 ± 0.32	86.98 ± 0.60
	MELR [111]	72.14 ± 0.51	87.01 ± 0.35
	COSOC [82]	73.57 ± 0.43	87.57 ± 0.10
	CNL [110]	73.42 ± 0.95	87.72 ± 0.75
	invariance-equivariance [112]	72.21 ± 0.90	87.08 ± 0.58
	CSEI [86]	73.76 ± 0.32	87.83 ± 0.59
	ASY ResNet12 (ours)	$\mathbf{74.31 \pm 0.22}$	$\mathbf{87.86 \pm 0.15}$
$36M$	S2M2R [67]	73.71 ± 0.22	$\mathbf{88.52 \pm 0.14}$
	EASY 3×ResNet12 (ours)	$\mathbf{74.71 \pm 0.22}$	88.33 ± 0.14

Table 2.3 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **CIFAR-FS** in **inductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	S2M2R [67]	63.66 ± 0.17	76.07 ± 0.19
	R2-D2 (+ens) [95]	76.51 ± 0.47	87.63 ± 0.34
	invariance-equivariance [112]	$\mathbf{77.87 \pm 0.85}$	$\mathbf{89.74 \pm 0.57}$
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	75.24 ± 0.20	88.38 ± 0.14
$36M$	S2M2R [67]	74.81 ± 0.19	87.47 ± 0.13
	EASY 3×ResNet12 (ours)	$\mathbf{76.20 \pm 0.20}$	$\mathbf{89.00 \pm 0.14}$

Table 2.4 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **CUB-FS** in **inductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	FEAT [109]	68.87 ± 0.22	82.90 ± 0.10
	ProtoNet [83]	66.09 ± 0.92	82.50 ± 0.58
	DeepEMD v2 [87]	$\mathbf{79.27 \pm 0.29}$	89.80 ± 0.51
	EASY 4×ResNet12$\left(\frac{1}{2}\right)$ (ours)	77.97 ± 0.20	$\mathbf{91.59 \pm 0.10}$
$36M$	S2M2R [67]	$\mathbf{80.68 \pm 0.81}$	90.85 ± 0.44
	EASY 3×ResNet12 (ours)	78.56 ± 0.19	$\mathbf{91.93 \pm 0.10}$

Table 2.5 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **FC-100** in **inductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	DeepEMD v2 [87]	46.60 ± 0.26	63.22 ± 0.71
	TADAM [108]	40.10 ± 0.40	56.10 ± 0.40
	ProtoNet [83]	41.54 ± 0.76	57.08 ± 0.76
	invariance-equivariance [112]	47.76 ± 0.77	$\mathbf{65.30 \pm 0.76}$
	R2-D2 (+ens) [95]	44.75 ± 0.43	59.94 ± 0.41
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	$\mathbf{47.94 \pm 0.19}$	64.14 ± 0.19
$36M$	EASY 3×ResNet12 (ours)	$\mathbf{48.07 \pm 0.19}$	64.74 ± 0.19

Table 2.6 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **MiniImageNet** in **transductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	TIM-GD [113]	73.90	85.00
	ODC [114]	77.20 ± 0.36	87.11 ± 0.42
	PEM$_n$E-BMS* [104]	80.56 ± 0.27	87.98 ± 0.14
	SSR [115]	68.10 ± 0.60	76.90 ± 0.40
	iLPC [116]	69.79 ± 0.99	79.82 ± 0.55
	EPNet [103]	66.50 ± 0.89	81.60 ± 0.60
	DPGN [117]	67.77 ± 0.32	84.60 ± 0.43
	ECKPN [118]	70.48 ± 0.38	85.42 ± 0.46
	Rot+KD+POODLE [119]	77.56	85.81
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	$\mathbf{82.31 \pm 0.24}$	$\mathbf{88.57 \pm 0.12}$
$36M$	SSR [115]	72.40 ± 0.60	80.20 ± 0.40
	fine-tuning(train+val) [120]	68.11 ± 0.69	80.36 ± 0.50
	SIB+E^{3}BM [121]	71.40	81.20
	LR+DC [91]	68.57 ± 0.55	82.88 ± 0.42
	EPNet [103]	70.74 ± 0.85	84.34 ± 0.53
	TIM-GD [113]	77.80	87.40
	PT+MAP [122]	82.92 ± 0.26	88.82 ± 0.13
	iLPC [116]	83.05 ± 0.79	88.82 ± 0.42
	ODC [114]	80.64 ± 0.34	89.39 ± 0.39
	PEM$_n$E-BMS* [104]	83.35 ± 0.25	$\mathbf{89.53 \pm 0.13}$
	EASY 3×ResNet12 (ours)	$\mathbf{84.04 \pm 0.23}$	89.14 ± 0.11

Table 2.7 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **CUB-FS** in **transductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	TIM-GD [113]	82.20	90.80
	ODC [114]	85.87	**94.97**
	DPGN [117]	75.71 ± 0.47	91.48 ± 0.33
	ECKPN [118]	77.43 ± 0.54	92.21 ± 0.41
	iLPC [116]	89.00 ± 0.70	92.74 ± 0.35
	Rot+KD+POODLE [119]	89.93	93.78
	EASY 4×ResNet12$\left(\frac{1}{2}\right)$ (ours)	$\mathbf{90.50 \pm 0.19}$	93.50 ± 0.09
$36M$	LR+DC [91]	79.56 ± 0.87	90.67 ± 0.35
	PT+MAP [122]	$\mathbf{91.55 \pm 0.19}$	93.99 ± 0.10
	iLPC [116]	91.03 ± 0.63	$\mathbf{94.11 \pm 0.30}$
	EASY 3×ResNet12 (ours)	90.56 ± 0.19	93.79 ± 0.10

Table 2.8 – 1-shot and 5-shot accuracy of state-of-the-art methodsand proposed solution on **FC-100** in **transductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	TADAM [108]	40.10 ± 0.40	56.10 ± 0.40
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	$\mathbf{54.47 \pm 0.24}$	$\mathbf{65.82 \pm 0.19}$
$36M$	SIB+E^{3}BM [121]	46.00	57.10
	fine-tuning (train) [120]	43.16 ± 0.59	57.57 ± 0.55
	ODC [114]	47.18 ± 0.30	59.21 ± 0.56
	fine-tuning (train+val) [120]	50.44 ± 0.68	65.74 ± 0.60
	EASY 3×ResNet12 (ours)	$\mathbf{54.13 \pm 0.24}$	$\mathbf{66.86 \pm 0.19}$

Table 2.9 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **CIFAR-FS** in **transductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	SSR [115]	76.80 ± 0.60	83.70 ± 0.40
	iLPC [116]	77.14 ± 0.95	85.23 ± 0.55
	DPGN [117]	77.90 ± 0.50	90.02 ± 0.40
	ECKPN [118]	79.20 ± 0.40	$\mathbf{91.00 \pm 0.50}$
	EASY 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ (ours)	$\mathbf{86.99 \pm 0.21}$	90.20 ± 0.15
$36M$	SSR [115]	81.60 ± 0.60	86.00 ± 0.40
	fine-tuning (train+val) [120]	78.36 ± 0.70	87.54 ± 0.49
	iLPC [116]	86.51 ± 0.75	90.60 ± 0.48
	PT+MAP [122]	$\mathbf{87.69 \pm 0.23}$	$\mathbf{90.68 \pm 0.15}$
	EASY 3×ResNet12 (ours)	87.16 ± 0.21	90.47 ± 0.15

Table 2.10 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **TieredImageNet** in **transductive** setting.

	Method	1-shot	5-shot
$\leq 12M$	PT+MAP [122]	85.67 ± 0.26	90.45 ± 0.14
	TIM-GD [113]	79.90	88.50
	ODC [114]	83.73 ± 0.36	$\mathbf{90.46 \pm 0.46}$
	SSR [115]	81.20 ± 0.60	85.70 ± 0.40
	Rot+KD+POODLE [119]	79.67	86.96
	DPGN [117]	72.45 ± 0.51	87.24 ± 0.39
	EPNet [103]	76.53 ± 0.87	87.32 ± 0.64
	ECKPN [118]	73.59 ± 0.45	88.13 ± 0.28
	iLPC [116]	83.49 ± 0.88	89.48 ± 0.47
	ASY ResNet12 (ours)	$\mathbf{83.98 \pm 0.24}$	89.26 ± 0.14
$36M$	SIB+E^{3}BM [121]	75.60	84.30
	SSR [115]	79.50 ± 0.60	84.80 ± 0.40
	fine-tuning (train+val) [120]	72.87 ± 0.71	86.15 ± 0.50
	TIM-GD [113]	82.10	89.80
	LR+DC [91]	78.19 ± 0.25	89.90 ± 0.41
	EPNet [103]	78.50 ± 0.91	88.36 ± 0.57
	ODC [114]	85.22 ± 0.34	91.35 ± 0.42
	iLPC [116]	$\mathbf{88.50 \pm 0.75}$	$\mathbf{92.46 \pm 0.42}$
	PEM$_n$E-BMS* [104]	86.07 ± 0.25	91.09 ± 0.14
	EASY 3×ResNet12 (ours)	84.29 ± 0.24	89.76 ± 0.14

2.4.2 Ablation study

The relative contributions of components in the proposed method is of interest here. To grasp them, we compare, for each dataset, the performance of various combinations in Table 2.11 in the inductive case, and Table 2.12 in the transductive case. Surprisingly, our full proposed methodology (EASY) is not always the best performing. We argue that for large datasets such as MiniImageNet and TieredImageNet, the considered ResNet12 backbones does not contain enough parameters. When reducing the number of parameters for ensemble solutions, the performance drop due to the reduction in size is not balanced

by the diversity of the multiple backbones. When all is said and done, only AS proves consistently beneficial to the accuracy.

2.4.3 Discussion

The proposed method achieves state-of-the-art performance on MiniImagenet by a fair margin in the inductive case. On TieredImagenet, only S2M2R outperforms EASY in the 5-shot setting. It might be due to TieredImagenet being the largest of the considered datasets and thus requiring more parameters to be trained efficiently and reducing the effectiveness of the proposed ensemble approach. The poor performance on CIFAR-FS can be explained by the small resolution of images in the dataset which works poorly with the augmented sample step. Our results in the inductive setting are overlapping with [112] on FC-100 dataset, however our method provides a narrower confidence intervals compared to other methods on the same benchmark, despite using the same number of tasks. In the transductive case, our method achieves competitive accuracies without any prior on the number of sample per classes. This is particularly important since several methods tend to fail when the number of samples per class is different, as shown in the supplementary material. Our explanation is that multiple methods tend to over-rely on this prior. This over-reliance concern was first raised by [123]. Overall, our method is easy to implement and requires minimal hyperparameter tuning compared to other competitive approaches.

2.5 Transductive tests with imbalanced settings

We also report performance in transductive setting when the number of query vectors is varying for each class and is unknown. We use the protocol described in [123]. We present the results in Tables 2.13-2.15. The Confidence Intervals of previously published methods were not reported by [123]. We show that our method clearly outperforms existing ones significantly.

Table 2.11 – Ablation study of the steps of proposed solution in **inductive** setting, for a fixed number of trainable parameters in the considered backbones. When using ensembles, we use 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ instead of a single ResNet12.

Dataset	E	AS	1-shot	5-shot
MiniImageNet			68.43 ± 0.19	83.78 ± 0.13
		✓	$\mathbf{70.84 \pm 0.19}$	85.70 ± 0.13
	✓		68.69 ± 0.20	84.84 ± 0.13
	✓	✓	70.63 ± 0.20	$\mathbf{86.28 \pm 0.12}$
CUB-FS			74.13 ± 0.20	89.08 ± 0.11
		✓	77.40 ± 0.20	$\mathbf{91.15 \pm 0.10}$
	✓		75.01 ± 0.20	89.38 ± 0.11
	✓	✓	$\mathbf{77.59 \pm 0.20}$	91.07 ± 0.11
CIFAR-FS			73.38 ± 0.21	87.42 ± 0.15
		✓	74.26 ± 0.21	88.16 ± 0.15
	✓		74.36 ± 0.21	87.82 ± 0.15
	✓	✓	$\mathbf{75.24 \pm 0.20}$	$\mathbf{88.38 \pm 0.14}$
FC-100			45.68 ± 0.19	62.78 ± 0.19
		✓	46.43 ± 0.19	64.16 ± 0.19
	✓		47.52 ± 0.19	63.92 ± 0.19
	✓	✓	$\mathbf{47.94 \pm 0.20}$	$\mathbf{64.14 \pm 0.19}$
TieredImageNet			72.52 ± 0.22	86.79 ± 0.15
		✓	$\mathbf{74.17 \pm 0.22}$	$\mathbf{87.81 \pm 0.14}$
	✓		72.14 ± 0.22	86.66 ± 0.15
	✓	✓	73.36 ± 0.22	87.37 ± 0.15

Table 2.12 – Ablation study of the steps of proposed solution in **transductive** setting for a fixed number of trainable parameters in the considered backbones. When using ensembles, we use 2×ResNet12$\left(\frac{1}{\sqrt{2}}\right)$ instead of a single ResNet12.

Dataset	E	AS	1-shot	5-shot
MiniImageNet			80.42 ± 0.23	86.72 ± 0.13
		✓	$\mathbf{83.02 \pm 0.23}$	88.36 ± 0.12
	✓		80.27 ± 0.23	87.45 ± 0.12
	✓	✓	82.31 ± 0.24	$\mathbf{88.57 \pm 0.12}$
CUB-FS			86.93 ± 0.21	91.53 ± 0.11
		✓	89.80 ± 0.20	93.12 ± 0.10
	✓		87.28 ± 0.21	91.89 ± 0.10
	✓	✓	$\mathbf{90.05 \pm 0.19}$	$\mathbf{93.17 \pm 0.10}$
CIFAR-FS			84.18 ± 0.23	89.56 ± 0.15
		✓	85.55 ± 0.23	90.07 ± 0.15
	✓		84.89 ± 0.22	89.60 ± 0.15
	✓	✓	$\mathbf{86.99 \pm 0.21}$	$\mathbf{90.20 \pm 0.15}$
FC-100			51.74 ± 0.23	65.39 ± 0.19
		✓	52.93 ± 0.23	$\mathbf{66.51 \pm 0.19}$
	✓		53.39 ± 0.23	65.71 ± 0.19
	✓	✓	$\mathbf{54.47 \pm 0.24}$	65.82 ± 0.19
TieredImageNet			82.32 ± 0.24	88.45 ± 0.15
		✓	$\mathbf{83.98 \pm 0.24}$	$\mathbf{89.26 \pm 0.14}$
	✓		81.48 ± 0.25	88.40 ± 0.15
	✓	✓	83.20 ± 0.25	88.92 ± 0.14

Table 2.13 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **MiniImageNet** in **imbalanced transductive** setting. (CC-BY)

	Method	1-shot	5-shot
$\leq 12M$	MAML [70]	47.6	64.5
	LR+ICI [124]	58.7	73.5
	PT+MAP [122]	60.1	67.1
	LaplacianShot [125]	65.4	81.6
	TIM [113]	67.3	79.8
	α-TIM [123]	67.4	82.5
	ASY ResNet12 (ours)	$\mathbf{75.65 \pm 0.25}$	$\mathbf{86.35 \pm 0.14}$
$36M$	PT+MAP [122]	60.6	66.8
	SIB [126]	64.7	72.5
	LaplacianShot [125]	68.1	83.2
	TIM [113]	69.8	81.6
	α-TIM [123]	69.8	84.8
	EASY 3×ResNet12 (ours)	$\mathbf{76.04 \pm 0.27}$	$\mathbf{87.23 \pm 0.15}$

Table 2.14 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **TieredImageNet** in **imbalanced transductive** setting. (CC-BY)

	Method	1-shot	5-shot
$\leq 12M$	Entropy-min [120]	61.2	75.5
	PT+MAP [122]	64.1	70.0
	LaplacianShot [125]	72.3	85.7
	TIM [113]	74.1	84.1
	LR+ICI [124]	74.6	85.1
	α-TIM [123]	74.4	86.6
	ASY ResNet12 (ours)	$\mathbf{78.15 \pm 0.27}$	$\mathbf{87.65 \pm 0.17}$
$36M$	Entropy-min [120]	62.9	77.3
	PT+MAP [122]	65.1	71.0
	LaplacianShot [125]	73.5	86.8
	TIM [113]	75.8	85.4
	α-TIM [123]	76.0	87.8
	EASY 3×ResNet12 (ours)	$\mathbf{78.46 \pm 0.28}$	$\mathbf{87.85 \pm 0.13}$

Table 2.15 – 1-shot and 5-shot accuracy of state-of-the-art methods and proposed solution on **CUB-FS** in **imbalanced transductive** setting. (CC-BY)

	Method	1-shot	5-shot
$\leq 12M$	PT+MAP [122]	65.1	71.3
	Entropy-min [120]	67.5	82.9
	LaplacianShot [125]	73.7	87.7
	TIM [113]	74.8	86.9
	α-TIM [123]	75.7	89.8
	ASY ResNet12 (ours)	**81.24 ± 0.27**	**87.27 ± 0.14**
$36M$	EASY 3×ResNet12 (ours)	**83.63 ± 0.25**	**92.35 ± 0.09**

2.6 Conclusion

In this chapter we introduced a simple *feature extractor* to perform few-shot classification in both inductive and transductive cases. We combined it with augmented samples and ensembling and showed its ability to reach state-of-the-art accuracies when deployed using simple classifiers on multiple standardized benchmarks. Our method even outperformed previous methods by a fair margin ($> 1\%$) in some cases.

2.7 What would we do differently now?

Since the beginning of this PhD, significant advancements have occurred in Few-Shot Learning (FSL). One major shift is the replacement of training custom feature extractors with the use of foundation models like visual CLIP and DINO v2, which offer strong, off-the-shelf representations for nearly all domains. The study in [9] serves as an excellent reference point for covering most of the methods developed in the field, though it provides a holistic comparison of various approaches rather than a detailed, step-by-step ablation of individual components. A valuable next step would be to break down these adaptation methods into individual components and conduct a thorough ablation study.

Additionally, current benchmarks often combine visual information with language data to enhance classifier performance [127, 128, 129]. An ablation study focused on the contribution of semantic class information would also be highly beneficial in understanding its impact.

Chapter 3

Assessing Accuracy Uncertainty while considering task inter-depedance

Contents

3.1 Introduction 69
3.2 Closed CIs vs. Open CIs 72
3.2.1 A mathematical description of the problem 72
3.2.2 Are OCIs larger than CCIs? An empirical study 75
3.2.3 Impact on Conclusiveness 77
3.3 Paired tests 77
3.3.1 Definitions 77
3.4 Sizing tasks to narrow OCIs 80
3.4.1 Mathematical derivation of $\mathrm{Var}(\bar{A})$ 81
3.4.2 Effect of S and N on Q^* 83
3.5 Benchmark Proposal 87
3.6 Related Work 90
3.7 Limitations 91
3.8 Conclusion 92
3.9 What would we do differently now? 92

3.1 Introduction

The content of this chapter is largely based on our paper *Oops, I Sampled it Again: Reinterpreting Confidence Intervals in Few-Shot Learning* [7] published in TMLR 2024.

As discussed earlier, FSL study has led to a proliferation of new methods and novel experimental protocols [130, 5, 4, 87]. Unlike conventional machine learning, which typically benchmarks methods using fixed training and validation splits, FSL faces distinct challenges due to its reliance on very small and thus often biased training datasets. In fact, the accuracy of FSL can vary significantly based on the choice of labeled training samples [131].

A common concern in Few-Shot Learning (FSL), similar to traditional machine learning, is identifying the best-performing methods. In FSL, the variability in performance based on the choice of labeled data has led practitioners to widely adopt the practice of aggregating statistics across a large number of artificially created tasks derived from one or a few datasets. The typical approach involves generating these few-shot tasks by randomly sampling the same dataset with replacement, allowing the same samples to appear in multiple tasks. By evaluating the performance across these numerous tasks, researchers can calculate the average accuracy and its confidence interval (CI) for each method, thus providing a statistically sound basis for comparing the effectiveness of different approaches.

		CLIP		DINO	
		NCM	FT	NCM	FT
CLIP	NCM		0 0 0	0 + +	**− 0 +**
	FT	− 0 0		0 + +	0 0 +
DINO	NCM	+ 0 0	+ 0 +		0 0 −
	FT	+ 0 0	+ 0 0	0 0 0	

Table 3.1 – Comparison of different methods for few-shot classification. Each entry in the table contains three elements: [**With Replacement (Closed), Without Replacement (Open), Paired Tests (PT)**]. Symbols + and − indicate significant positive and negative differences, respectively, between the method in the row and the method in the column, while 0 denotes inconclusive results. The results are based on the DTD test split (bottom-left triangle) and the Traffic Signs split (top-right triangle) of MetaDataset, with task sampling set to 5 shots, 5 ways, and 15 queries. **Pay attention to the inversion in bold**. NCM (Nearest Class Mean), FT (Fine-tune) with CLIP and DINO as feature extractors. (CC-BY)

By allowing the same samples to appear across multiple tasks, the computed CIs reflect the randomness of the sampler rather than the data itself. These CIs are based on the standard Lindeberg-L'evy Central Limit Theorem (CLT), which requires the underlying random variables to be independent and identically distributed (IID) for the CIs to be statistically valid. This means that the CIs currently reported should be seen as representing the likely range of outcomes if the

experiment were repeated using *exactly the same data*. We refer to these as **Closed CIs (CCIs)** throughout this chapter. However, what is often more relevant in machine learning is the range of outcomes if the experiment were repeated with data drawn from the same *underlying distribution*, which we call **Open CIs (OCIs)**. Although OCIs could be derived by sampling tasks without replacement, this approach severely limits the number of distinct tasks that can be generated from a given dataset. This constraint is particularly problematic with smaller datasets, leading to potentially larger CIs and less decisive comparisons between methods.

This chapter aims to underscore the importance of this key factor when calculating CIs. We introduce approaches to tackle this challenge and achieve meaningful comparisons while still incorporating data randomness. Our strategies involve: a) **Paired Tests (PT)**, where different methods are assessed on identical sets of generated tasks, and b) ensuring tasks are appropriately sized. The discussion throughout the chapter centers on the specific context of few-shot classification in vision, which remains the most widely studied area within FSL research.

Our investigation using Open Confidence Intervals (OCIs) reveals that conclusions drawn from the classical approach in few-shot learning can often be inconsistent. Notably, we find that some methods previously reported as statistically superior to others are actually indistinguishable when evaluated using OCIs, and the reverse can also be true. Furthermore, we demonstrate cases where using Closed Confidence Intervals (CCIs) leads to statistically significant conclusions that directly contradict those obtained through Paired Tests (PT). An illustrative example is provided in Table 3.1, where we compare various methods for few-shot classification based on their feature extractors (CLIP [58] or DINO [132]) and adaptation techniques (Logistic Regression (LR), Nearest Class Centroid (NCM), or Fine-Tuning (FT)). The table presents three conclusions for each pair of model and method combinations: the first from the methodology in [9], which uses the common approach of calculating CCIs; the second from OCIs; and the third based on PT. A result where the row method outperforms the column method is indicated by +, while 0 signifies no conclusive difference, and – indicates the column method outperforms the row method. The upper triangular values in blue correspond to the Traffic Sign test split, while the red lower triangle pertains to the DTD test split from the Metadataset benchmark by [8]. A particularly striking example emerges from the Traffic Signs dataset. When evaluated with replacement tests, CLIP with the NCM adapter appears to underperform compared to DINO with Fine-Tuning, yet this result is reversed when using paired tests. This starkly

highlights the importance of carefully choosing measurement methods, as relying on the wrong approach can lead to significant misinterpretations of the results.

Here are the main contributions of this work. They were first introduced in our paper [7].

— We emphasized the critical role of considering data replacement when calculating Confidence Intervals (CIs) for comparing FSL methods. Our study demonstrated how the transition from closed to open CIs affects CI ranges on standard off-the-shelf vision datasets.

— Through paired evaluations, where multiple methods are assessed on the same set of generated tasks, we showed that this approach leads to more frequent conclusive comparisons than relying solely on OCIs.

— We explored strategies for optimizing task generation from a given dataset, considering its size and the number of classes, to achieve narrower CIs and facilitate more definitive comparisons between methods. This resulted in a benchmark tailored for few-shot image classification.

3.2 Closed CIs vs. Open CIs

In this section, we aim to provide a clearer quantification of the differences between CCIs and OCIs. To achieve this, we introduce relevant notations and describe algorithms for task sampling. We then present a theoretical analysis to highlight the distinctions between CCIs and OCIs, followed by an empirical comparison of their ranges on real datasets.

3.2.1 A mathematical description of the problem

Standard Evaluation and Notations

The standard evaluation method in the field of few-shot classification is outlined in Algorithm 1. A few-shot classification task, denoted as $\mathcal{T} = (\mathcal{K}, \mathcal{S}, \mathcal{Q})$, consists of a set of classes $\mathcal{K}$, a support set $\mathcal{S} = \mathcal{S}_{c\in\mathcal{K}}$, and a query set $\mathcal{Q} = \mathcal{Q}_{c\in\mathcal{K}}$, where $\mathcal{S}_c$ and $\mathcal{Q}_c$ represent the support and query examples for each class $c \in \mathcal{K}$.

Let $K = |\mathcal{K}|$ represent the number of *ways* (i.e., the number of classes in a few-shot task), $S = |\mathcal{S}c|$ denote the number of *shots* per class, and $Q = |\mathcal{Q}_c|$ the number of *queries* per class (assuming balanced classes for simplicity). Typically, few-shot evaluation involves

constructing numerous tasks from a larger evaluation dataset.

An evaluation dataset, $\mathcal{D} = (\mathcal{C}, \mathcal{X})$, consists of a set of classes $\mathcal{C}$ and examples for all classes, $\mathcal{X} = \mathcal{X}_{c \in \mathcal{C}}$. Here, $C = |\mathcal{C}| \geq K$ represents the number of classes, and $N = |\mathcal{X}_c| \gg S + Q$ denotes the number of examples per class.

As noted in the introduction of this chapter, the prevailing approach to CI computation assumes that $\mathcal{D}$ is fixed and deterministic, rather than probabilistic.

Algorithm 1 Predominant evaluation algorithm (CC-BY)

procedure EVALUATE($T, K, S, Q, \mathcal{C}, \{\mathcal{X}_c\}_{c \in \mathcal{C}}$) ▷ T tasks, K ways, S shots, Q queries, set of classes C, set of data samples $\{\mathcal{X}_c\}_{c \in \mathcal{C}}$
  for $t = 1, \ldots, T$ **do**
    $\mathcal{K} \leftarrow$ take(K, shuffle($\mathcal{C}$))
    for $c \in \mathcal{K}$ **do**
      $\mathcal{S}_c, \mathcal{Q}_c \leftarrow$ split(S, take($S + Q$, shuffle($\mathcal{X}_c$)))
    end for
    $f_{\mathcal{S}} \leftarrow$ FEWSHOTLEARN($\mathcal{S}$) ▷ $\mathcal{S} := \{\mathcal{S}_c\}_{c \in \mathcal{K}}$
    $\bar{A}_t \leftarrow \frac{1}{KQ} \sum_{c \in \mathcal{K}} \sum_{x \in \mathcal{Q}_c} \mathbb{1}[f_{\mathcal{S}}(x) = c]$
  end for
  $\bar{A} \leftarrow$ MEAN(A) ▷ MEAN(A) := $\frac{1}{T} \sum_{t=1}^{T} A_t$
  $\sigma_A \leftarrow \sqrt{\text{VAR}(A)}$ ▷ VAR(A) := $\frac{1}{T-1} \sum_{t=1}^{T} (A_t - \bar{A})^2$
  $\sigma_{\bar{A}} \leftarrow \sigma_A / \sqrt{T}$
  return $\bar{A} \pm 1.96\sigma_{\bar{A}}$ ▷ $1.96\sigma_{\bar{A}} = r(p_{limit} = 95\%)\sigma_{\bar{A}}$
end procedure

The standard few-shot task sampler generates T random tasks with K ways, S shots, and Q queries from a dataset $\mathcal{D}$, as described in Algorithm 1. This process introduces additional randomness beyond the dataset itself, specifically in the selection of classes and examples. Let $\mathcal{T}_t = (\mathcal{K}_t, \mathcal{S}_t, \mathcal{Q}_t)$ for $t = 1, \ldots, T$ represent the sampled tasks.

The accuracy for each task is computed as follows:

$$A_t = \frac{1}{KQ} \sum_{c \in \mathcal{K}_t} \sum_{x \in \mathcal{Q}_{tc}} \mathbb{1}[f_{\mathcal{S}_t}(x) = c], \tag{3.1}$$

where f is the model being evaluated, conditioned on the support set $\mathcal{S}_t$.

The metric typically reported is the average accuracy across these multiple tasks. This

is the focus of the next paragraph.

Computing Confidence Intervals

We calculate the accuracy A_t for the method on each task and then compute the mean accuracy across all tasks as $\bar{A} = \frac{1}{T}(A_1 + \cdots + A_T)$. This leads to the following formula for the variance:

$$\text{Var}[\bar{A}] = \frac{\text{Var}[A_1] + \cdots + \text{Var}[A_T]}{T^2} = \frac{\text{Var}[A]}{T} \ . \tag{3.2}$$

Assuming a sufficiently large sample size and that the mean $\bar{A}$ follows a normal distribution as per the Central Limit Theorem, the 95% confidence interval can be calculated as $\bar{A} \pm 1.96\sigma_{\bar{A}}$, where $\sigma_{\bar{A}}$ is the standard error (the standard deviation of the sample mean), given by the formula:

$$CI = 1.96\sigma_{\bar{A}} = 1.96\frac{\sigma_A}{\sqrt{T}} \ . \tag{3.3}$$

It's important to note that when dealing with a very small number of tasks, Student's t-distribution can be used instead. Additionally, while we used 95% confidence intervals (CIs) as an example—since this is a common choice in the literature—this is ultimately an arbitrary selection. For greater generality, we introduce a probability p_{limit} for all theoretical considerations moving forward, but we will continue using the 95

As the number of tasks T increases, it becomes inevitable that many tasks will reuse examples, since they are constructed independently *with* replacement. Consequently, as T grows large, the variance of the sample mean will converge to the *conditional* variance $\text{Var}[\bar{A} \mid \mathcal{D}]$, and the resulting confidence interval will reflect the likely range of outcomes if the experiment were repeated with a different set of randomly selected tasks from the *same dataset*. This means that such an interval does not provide any insight into how well the method would generalize to a distribution.

Conversely, if T is kept small enough, the reuse of examples will be minimal, and the assumption of independence might approximately hold, though this will likely result in a significantly wider confidence interval.

We now empirically evaluate the differences between closed (tasks sampled with replacement) and open (tasks sampled without replacement) confidence intervals using real datasets.

3.2.2 Are OCIs larger than CCIs? An empirical study

In contrast to Algorithm 1, the task sampling process without replacement is described in Algorithm 4 (see Appendix). This algorithm explicitly employs a Student's t-distribution estimator, anticipating that for some small datasets, only a limited number of independent tasks can be generated. In this approach, the total number of tasks T is determined by a specific stopping condition—namely, the exhaustion of the dataset—aimed at minimizing the range of the resulting confidence intervals (CIs). Since each sample is used only once, classes and examples can be considered as drawn independently and identically distributed (IID) from underlying distributions $p(C)$ and $p(X \mid C)$. This is why Open Confidence Intervals (OCIs) account for the inherent randomness in the data.

Algorithm 1 samples tasks with replacement and computes CCIs with Equation 3.3 while Algorithm 4 samples tasks without replacement and uses the student's distribution to compute OCIs.

For our experiments, we utilized datasets from the MetaDataset Benchmark, as described in [8]. This benchmark includes 10 datasets, of which we employed 9, excluding Imagenet, to focus on cross-domain results in line with recent trends in the literature [133]. The datasets we used include Omniglot (handwritten characters), Aircraft, CUB (birds), DTD (textures), Fungi, VGG Flowers, Traffic Signs, Quickdraw (crowd-sourced drawings), and MSCOCO (common objects) [134, 135, 136, 137, 138, 139, 140, 141, 142].

[9] presents few-shot accuracies for 2000 tasks with 5-shots, 5-ways, and 15 queries, in a comprehensive table covering various works on the MetaDataset datasets. The only deviation in our study is the adoption of a $T = 600$ setting, which is more prevalent in existing literature. If CCIs are found to be narrower than OCIs with this smaller T, the difference will be even more pronounced with $T = 2000$ tasks, as shown in Equation 3.3. Our primary reference for methods and models is the detailed compilation by [9], which serves as the foundational starting point for our experiments.

Our findings, summarized in Table 3.2, present results across various few-shot methods and datasets. First, we observe that CCIs exhibit notable consistency, stemming from the fixed number of tasks set at $T = 600$, which contrasts with the variability seen in OCIs. Interestingly, CCIs are significantly narrower than OCIs for smaller datasets such as Aircraft and DTD. On the other hand, for larger datasets like Quickdraw, CCIs become broader than OCIs due to $T = 600$ being insufficient to fully deplete the dataset. Specifically, the test splits for Aircraft and DTD contain 1,500 and 840 samples, respec-

tively, while MSCOCO and Quickdraw have much larger test splits with 152,000 and 7.7 million samples, respectively. Across various datasets, models, and methods, CCIs are on average 3.8 times narrower than OCIs. These results underscore the critical importance of accurately interpreting Confidence Intervals, as the dramatic differences between OCI and CCI ranges can lead to conflicting conclusions if misinterpreted.

Additionally, we observe that in cases where methods achieve near-perfect accuracy—such as adaptation methods using CLIP (as opposed to DINO) on the CUB dataset—both types of CIs tend to narrow. This is due to accuracy saturation at 100%, which reduces the standard deviation of the accuracies.

		Model	CLIP		DINO	
		Sampling	W. Repl.	W/O. Repl.	W. Repl.	W/O. Repl.
Dataset Name	Dataset Size	Method				
DTD	840	LR	79.59 ± 0.52	84.00 ± 4.50	83.29 ± 0.51	86.10 ± 4.66
		FT	76.87 ± 0.56	80.76 ± 5.60	81.82 ± 0.51	84.19 ± 5.76
VGG Flower	1,425	LR	98.30 ± 0.23	99.39 ± 0.84	97.48 ± 0.26	97.58 ± 2.07
		FT	98.33 ± 0.23	99.27 ± 0.93	97.13 ± 0.29	97.45 ± 1.98
Aircraft	1,500	LR	75.79 ± 0.85	69.90 ± 5.59	59.04 ± 0.93	53.62 ± 5.66
		FT	74.70 ± 0.85	68.95 ± 5.82	54.58 ± 0.94	49.81 ± 5.17
CUB	1,770	LR	96.85 ± 0.28	97.24 ± 1.88	92.06 ± 0.43	89.69 ± 4.03
		FT	96.68 ± 0.29	97.07 ± 1.83	89.16 ± 0.52	87.47 ± 4.64
Omniglot	13,180	LR	90.55 ± 0.49	91.04 ± 0.89	93.67 ± 0.38	94.12 ± 0.79
		FT	92.06 ± 0.48	92.83 ± 0.91	94.70 ± 0.36	95.09 ± 0.70
Fungi	13,463	LR	70.86 ± 0.88	74.78 ± 1.89	77.26 ± 0.75	81.64 ± 1.41
		FT	68.03 ± 0.95	71.87 ± 1.96	74.02 ± 0.83	78.88 ± 1.68
Traffic Signs	39,252	LR	82.99 ± 0.87	77.02 ± 0.92	83.63 ± 0.92	75.58 ± 1.17
		FT	83.64 ± 0.85	76.69 ± 0.96	84.20 ± 0.92	74.93 ± 1.23
MSCOCO	151,545	LR	72.27 ± 0.78	67.97 ± 0.63	76.05 ± 0.72	72.02 ± 0.60
		FT	70.22 ± 0.79	65.97 ± 0.64	75.18 ± 0.75	71.19 ± 0.61
Quickdraw	7,710,295	LR	75.79 ± 0.66	75.54 ± 0.36	74.54 ± 0.68	74.07 ± 0.38
		FT	76.18 ± 0.69	75.93 ± 0.38	74.10 ± 0.70	73.61 ± 0.39

Table 3.2 – Accuracies and associated CIs of methods on different datasets with and without replacement in the sampling of tasks. The dataset size corresponds to the number of images in each test split. FT and LR respectively stand for Fine-tune and Logistic Regression. (CC-BY)

In the following sections, we explore the conclusiveness of comparative studies when employing either Closed Confidence Intervals (CCIs) or Open Confidence Intervals (OCIs).

3.2.3 Impact on Conclusiveness

First, we will review how confidence intervals are utilized to draw conclusions when comparing methods. Suppose we have two variables of interest, x_1 and x_2, with their respective p_{limit} confidence intervals (a generalized version of 95% confidence intervals) given by $[\bar{x}_1 - \delta_1, \bar{x}_1 + \delta_1]$ and $[\bar{x}_2 - \delta_2, \bar{x}_2 + \delta_2]$. To conclude that x_1 is smaller than x_2, we proceed as follows: if the two intervals do not overlap, and $\bar{x}_1 + \delta_1 < \bar{x}_2 - \delta_2$, then:

$$P\left(x_1 < x_2\right) > P\left(x_1 < \bar{x}_1 + \delta_1 \wedge x_2 > \bar{x}_2 - \delta_2\right) = \left(1 - \frac{1 - p_{limit}}{2}\right)^2 > p_{limit} \ , \tag{3.4}$$

where the $(1 - p_{limit})/2$ term arises from the symmetry of the Gaussian distribution.

Given this context, the results presented in Table 3.2 can lead to varying conclusions based on whether Closed Confidence Intervals (CCIs) or Open Confidence Intervals (OCIs) are used. For instance, using the DTD dataset, CCIs facilitate clear comparisons between different backbones and methods, whereas OCIs result in inconclusive comparisons due to overlapping intervals. This discrepancy should not be viewed as a contradiction but rather as an illustration of different paradigms for comparison. CCIs assess methods as if they were applied to the same data, while OCIs evaluate them in terms of the underlying distribution. To enhance the conclusiveness of method comparisons, we proposed [7] two approaches: paired tests, discussed in Section 3.3, and task sizing, covered in Section 3.4.

3.3 Paired tests

3.3.1 Definitions

To mitigate the range of Open Confidence Intervals (OCIs), we suggest employing paired tests. As noted in the introduction, FSL tasks vary greatly in difficulty, leading to significant variance in accuracy across tasks. Interestingly, a task that is considered *hard* for method A often presents a similar level of difficulty for method B. This correlation in task difficulty between different methods, identified by [131], is supported by our findings. As shown in Figure 3.1, which depicts accuracies on tasks generated from the Traffic Sign dataset using two different feature extractor and adaptation method combinations, there is a notable correlation of 0.675 between the two methods. This strong correlation underscores the potential for reducing accuracy variance due to task sampling by utilizing

paired testing.

Formally, let us define $\Delta_t = A_t - B_t$ as the accuracy difference between method A and method B on task t. The mean difference across tasks is given by $\bar{\Delta} = \frac{1}{T}\sum_{t=1}^{T}\Delta_t$. While the mean of the differences is simply the difference between the means, $\mathbb{E}[\bar{\Delta}] = \mathbb{E}[\bar{A}] - \mathbb{E}[\bar{B}]$, the variance of this difference can be significantly reduced if the accuracies are positively correlated across tasks.

$$\mathrm{Var}[\bar{\Delta}] = \mathrm{Var}\left[\frac{1}{T}\sum_{t=1}^{T}\Delta_t\right] = \frac{1}{T}\mathrm{Var}\left[A_t - B_t\right] \tag{3.5}$$

since $\mathrm{Var}[X - Y] = \mathrm{Var}[X] + \mathrm{Var}[Y] - 2\,\mathrm{Cov}(X, Y)$.

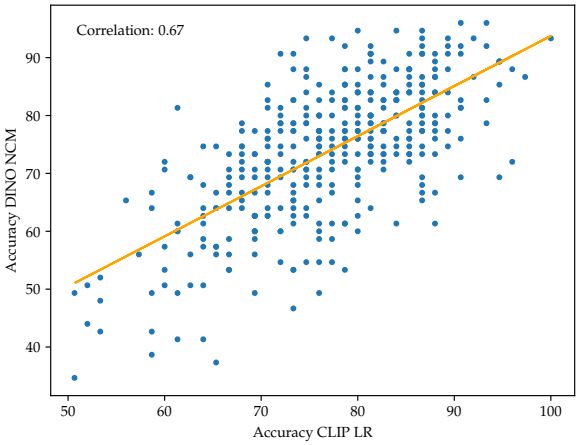


Figure 3.1 – Scatter plot of task accuracies for two different combinations of feature extractor and adaptation methods on the Traffic Signs benchmark. (CC-BY)

The reduced variance of Δ_t compared to A_t results in a correspondingly narrower confidence interval, as specified in Equation 3.3. Consequently, this can lead to situations where two methods show significant differences when analyzed using paired testing, even if no significant differences are observed when directly comparing the accuracies.

We conducted experiments comparing various methods to fine-tuning (FT), with the results presented in Table 3.3. Each row in the table corresponds to a specific dataset and feature extractor, while each column represents a combination of an adaptation method and a feature extractor. The table displays two sets of conclusions: one based on direct accuracy comparisons and the other based on paired tests. It is important to note that in all cases, we utilized Open Confidence Intervals (OCIs), meaning that tasks were sampled without replacement.

Furthermore, paired tests never yield conclusions that contradict those obtained from direct accuracy comparisons. This consistency stems from the previously discussed properties of the mean of differences. The primary difference between the two methods lies in their capacity to draw conclusive comparisons.

The fine-tuning adaptation method is used as the baseline in Table 3.3 primarily due to its significant computational cost. This cost arises from the update the all weights in the feature extractor and the head for each task. The goal is to assess whether fine-tuning

offers superior performance compared to more cost-effective methods. In comparative analyses focusing solely on the same feature extractor and excluding comparisons between CLIP and DINO, fine-tuning often either performs worse than other methods or yields inconclusive results. Specifically, FT outperforms other methods in only 4 out of 36 cases, while it underperforms in 14 cases. All remaining instances are inconclusive. Therefore, considering the substantial computational overhead, FT does not appear to offer a distinct advantage. It is also important to note that the effectiveness of fine-tuning is highly dependent on hyperparameters, meaning that any conclusions about this method are contingent upon a specific set of hyperparameters [143].

For the nine datasets under consideration, we conducted a total of 135 unique comparisons, focusing on distinct pairs of (model, method) across two models and three methods. Out of these, 57 comparisons yielded conclusive results using direct comparison with Open Confidence Intervals (OCI), whereas 94 comparisons were conclusive when using paired tests. This distribution is illustrated in Figure 3.6.

Table 3.1 demonstrates that the three methods for computing Confidence Intervals (CIs) lead to differing assessments of significance between two methods, as illustrated for the DTD and Traffic Signs datasets. Among the 114 cases where comparisons with replacement yielded conclusive results, approximately 23% (or 27 instances) show a pattern where an initially significant comparison becomes non-conclusive under sampling without replacement, but is conclusive again when paired tests are applied. Conversely, in around 11% of the cases (or 12 instances) where a comparison was previously considered significant, the paired test does not confirm conclusiveness.

A particularly striking example of inversion is observed in a case where a method initially deemed significantly more accurate than another was found to be significantly less accurate when using a paired test. This reversal occurred in the comparison of Fine-tuning (FT) with DINO versus Nearest Class Centroid (NCM) with CLIP features on the Traffic Signs dataset. This instance illustrates that a method may significantly outperform another on a specific dataset but underperform when evaluated across the entire distribution. It underscores that the dataset can be an specific instance that favors one method. This example highlights the critical importance of careful interpretation of Confidence Intervals (CIs).

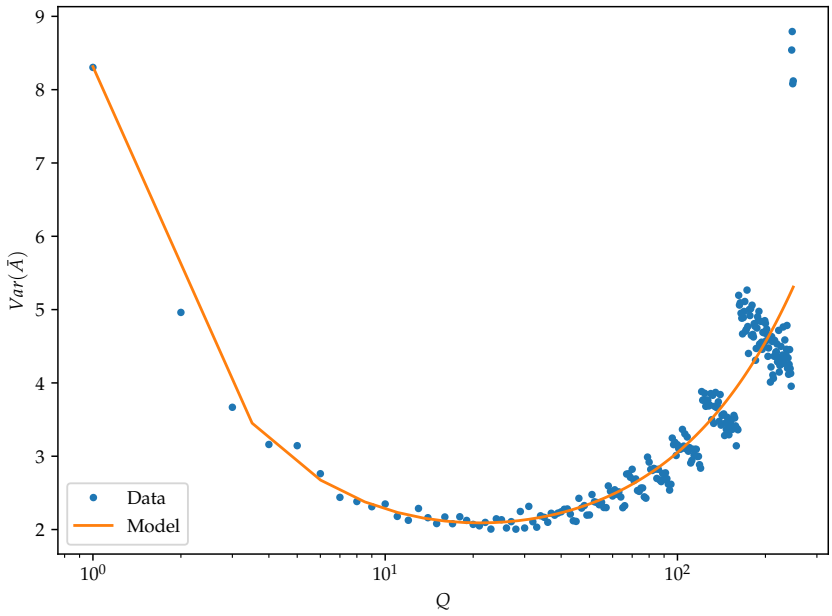


Figure 3.2 – Variance of the average accuracy versus the number of queries with synthetic data. The two classes are represented as 1D Gaussians $\mathcal{N}(-1, 1)$ and $\mathcal{N}(1, 1)$. The dataset size is $N = 1000$ (500 samples per class). Tasks are sampled according to Algorithm 4, with the number of shots fixed at 5. We fit this data using the model described in Equation 3.9 and observe a strong correspondence between the model and our experimental results. (CC-BY)

3.4 Sizing tasks to narrow OCIs

A pertinent issue that arises from the previous discussion is how to size tasks when performing sampling without replacement, with the aim of reducing the range of obtained CIs. In this context, we fix the size of the support set at $K \cdot S$, leaving the number of queries per task and per class Q as the variable to adjust. Increasing the number of queries will inevitably reduce the total number of tasks we can construct, as shown in the following equation. Assuming a balanced dataset, we can estimate the number of tasks T that can be sampled by fully utilizing the dataset:

$$T \approx \left\lfloor \frac{|\mathcal{D}|}{|\mathcal{T}|} \right\rfloor \approx \left\lfloor \frac{CN}{K(Q+S)} \right\rfloor , \tag{3.6}$$

where $|\mathcal{T}|$ represents the total number of samples in each task, including both the support and query sets.

Increasing Q not only reduces the number of tasks but also impacts $\sqrt{\text{Var}(A_t)}$, which directly influences the CI. Thus, a trade-off emerges between the number of queries and the number of tasks that can be generated, making it challenging to minimize OCIs effectively for any given dataset.

When $\bar{A}$ is measured with a small T (which results in a large Q), the CI ranges become quite broad, as outlined in Equation 3.3. Conversely, setting $Q = 1$ enables the generation of numerous tasks (large T) but introduces significant variance because class accuracy tends to be either 0% or 100%. To address this, our objective is to find the optimal number of queries, denoted as Q^*, that minimizes the variance of the average accuracy $\text{Var}(\bar{A})$ and thus the width of the CIs. We will first prove mathematically that such an optimal Q^* exists by deriving the expression for $\text{Var}(\bar{A})$.

3.4.1 Mathematical derivation of $\text{Var}(\bar{A})$

Suppose tasks are drawn IID *without* replacement, we write the variance of $\bar{A}$ as:

$$\text{Var}\left(\bar{A}\right) = \frac{1}{T}\text{Var}_t\left(A_t\right), \tag{3.7}$$

with A_t the accuracy for an arbitrary task t. By definition of the variance,

$$\text{Var}\left(A_t\right) = \mathbb{E}[(A_t)^2] - (\mathbb{E}[A_t])^2. \tag{3.8}$$

For a given support set, the expected accuracy for some class c is denoted $\mu_{t,c}$.

$$\mu_{t,c} \triangleq \mathbb{E}_{\mathcal{Q}_t}(\mathbb{1}[f_{\mathcal{S}_t}(x) = c] \mid \mathcal{S}_t).$$

Then the expectation of A_t becomes

$$\mathbb{E}[A_t] = \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c}],$$

Along the same lines, we can derive $\mathbb{E}[(A_t)^2]$,

$$\mathbb{E}[(A_t)^2] = \mathbb{E}_{\mathcal{S}_t}\mathbb{E}_{\mathcal{Q}_t}[(A_t)^2|\mathcal{S}_t].$$

For a fixed S_t, $\mathbb{1}[f_{\mathcal{S}_t}(x) = c]$ and $\mathbb{1}[f_{\mathcal{S}_t}(x') = c']$ are not independent and their distri-

bution will depend on the classes c and c'. Using Equation 3.1, we obtain .

$$\mathbb{E}[(A_t)^2] = \frac{1}{(KQ)^2} \sum_c \sum_{c'} \sum_x \sum_{x'} \mathbb{E}_{\mathcal{S}_t} \mathbb{E}_{\mathcal{Q}_t} [\mathbb{1}[f_{\mathcal{S}_t}(x) = c] \mathbb{1}[f_{\mathcal{S}_t}(x') = c']].$$

We now separate cases where $x = x'$ from cases where $c = c'$ and finally cases where both are different.

$$\mathbb{E}[(A_t)^2] = \frac{1}{KQ} \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c}] + \frac{Q-1}{KQ} \mathbb{E}_{\mathcal{S}_t}[(\mu_{t,c})^2] + \frac{1}{K^2} \sum_c \sum_{c' \neq c} \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c} \mu_{t,c'}].$$

Using Equation 3.8, we find:

$$\mathrm{Var}\,(A_t) = \frac{1}{KQ} \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c}] + \frac{Q-1}{KQ} \mathbb{E}_{\mathcal{S}_t}[(\mu_{t,c})^2] + \frac{1}{K^2} \sum_c \sum_{c' \neq c} \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c} \mu_{t,c'}] - (\mathbb{E}_{\mathcal{S}_t}[\mu_{t,c}])^2.$$

Let us define some parameters,

$$m_1 \triangleq \mathbb{E}[A_t] = \frac{1}{K} \sum_c \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c}],$$

$$m_2 \triangleq \frac{1}{K} \sum_c \mathbb{E}_{\mathcal{S}_t}[(\mu_{t,c})^2],$$

and

$$m_3 \triangleq \frac{1}{K^2} \sum_c \sum_{c' \neq c} \mathbb{E}_{\mathcal{S}_t}[\mu_{t,c} \mu_{t,c'}].$$

We get that

$$\mathbb{E}[(A_t)^2] = \frac{1}{KQ} m_1 + \frac{Q-1}{KQ} m_2 + m_3.$$

This gives

$$\mathrm{Var}\,(A_t) = \frac{1}{KQ} m_1 + \frac{Q-1}{KQ} m_2 + m_3 - (m_1)^2.$$

Then, using Equation 3.6 (removing the rounding) and 3.7, we approximate:

$$\mathrm{Var}\left(\bar{A}\right) = \frac{K}{NC}(\alpha Q + \frac{\beta}{Q} + \gamma), \tag{3.9}$$

with $\alpha = \frac{m_2}{K} + m_3 - (m_1)^2, \beta = \frac{S}{K}(m_1 - m_2)$ and $\gamma = \frac{m_1}{K} - \frac{m_2}{K} + \frac{S}{K} m_2 + S(m_3 - (m_1)^2)$.

First, let us notice that $\beta > 0$ since for $\mu \in [0, 1]$, $\mu^2 < \mu$.

These parameters are difficult to estimate in particular when dealing with real datasets

and methods. If $\alpha \leq 0$, then $\text{Var}\left(\bar{A}\right)$ is decreasing as a function of Q since $Q \in \mathbb{N}$. In the following, we focus only on cases where $\alpha > 0$. This choice is supported by empirical evidence, which we will present later, indicating a U-shaped relationship between the variance of $\bar{A}$ and Q for a certain range of S. Assuming this, $\text{Var}\left(\bar{A}\right)$ reaches its minimum at $Q^* = \sqrt{\frac{\beta}{\alpha}}$. Next, we study what this entails as S and N vary.

3.4.2 Effect of S and N on Q^*

Based on the definitions of α and β provided in Equation 3.9, we determine that Q^* increases with S and remains constant with respect to N. In this section, we provide empirical evidence demonstrating that these theoretical findings hold true in practical applications with real datasets.

We proposed [7] investigating the variance model of $\bar{A}$ in relation to Q using a simplified 1D sample representation. In our approach, we model two class distributions as Gaussians, $\mathcal{N}_i = \mathcal{N}(\mu_i, \sigma_i)$ for $i \in 1, 2$. We generate an artificial balanced dataset of size N. Tasks are sampled from this dataset until it is exhausted, following the process described in Algorithm 4, with K and C set to 2. Using the NCM classifier, we calculate the accuracies for these tasks to determine the average accuracy $\bar{A}$. This process of creating synthetic datasets from Gaussians and measuring $\bar{A}$ is repeated multiple times, yielding a set of $\{\bar{A}_j\}_j$ for specific parameters $S, Q, N, \mathcal{N}_1, \mathcal{N}_2$. We then compute the empirical variance $\text{Var}(\bar{A})$ based on these results.

In Figure 3.2, we present the measured $\text{Var}(\bar{A})$ as a function of Q. The datasets consist of 1000 samples evenly distributed between two classes, with the number of shots set to $S = 5$. The model described in Equation 3.9 fits the data with high accuracy. The discretization effect observed at high Q is attributed to the limited number of tasks. We will next explore how S and N influence $\text{Var}(\bar{A})$ and compare these findings with our synthetic data experiments.

As S increases, the curve's minimum shifts from $Q^* = 1$ toward $Q^* \to +\infty$, as illustrated in Figure 3.3. This trend confirms the predictions of our model. When $S = 1$, setting $Q^* = 1$ results in two notable effects: (a) the high variance of A_t due to the small support and query sets increases the variance of $\bar{A}$, and (b) a low Q allows for a significantly larger T, thereby reducing the overall variance of $\bar{A}$, as described in Equation 3.3. In contrast, for $S \geq 20$, the scenario effectively becomes one of classical transfer learning, where the narrowest confidence interval is achieved with a single task that has a large

support and query set.

We also identify a third regime where, for intermediate values of S, Q^* is nontrivial. This regime is exemplified by the $S = 5$ case shown in Figure 3.3.

Regarding the effect of increasing N, Equation 3.9 suggests that Q^* should remain unaffected. Indeed, Figure 3.3 shows only a minor and likely negligible shift in Q^* with increasing N. We will now investigate how these findings apply to real-world datasets.

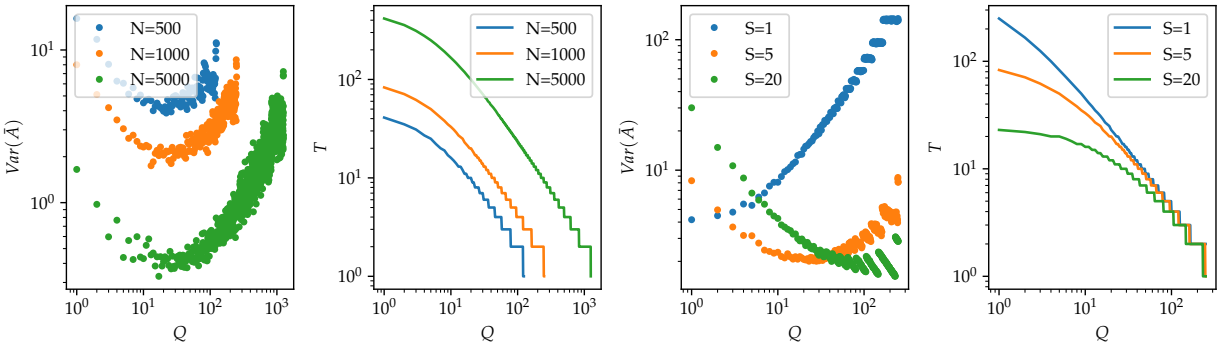


Figure 3.3 – Variance of the average accuracy $\bar{A}$ and the number of tasks T across varying settings of S and N, based on synthetic datasets with two 1D Gaussian distributions, $\mathcal{N}(-1, 1)$ and $\mathcal{N}(1, 1)$. The left pair of graphs illustrates results with a fixed number of shots ($S = 5$), whereas the right pair of graphs presents results for a constant sample size in the synthetic dataset ($N = 1000$). (CC-BY)

Real Dataset Experiments

We now examine the results from real image datasets, which often have imbalanced class distributions and varying numbers of classes. Our goal is to verify whether the findings from synthetic data apply to these more complex scenarios.

Our analysis confirms that Q^* remains independent of dataset size. However, a larger dataset size increases T, which consequently scales the confidence interval ($CI_{95\%}$). Consistent with synthetic data observations, we see a discretization of the confidence interval at high Q values, corresponding to a lower number of tasks (T).

The results across different shot settings (1, 5, and 10 shots) are consistent with synthetic data patterns, as shown in Figure 3.4. For a 1-shot setting, the optimal number of queries, Q, is 1. For 5-shot and 10-shot settings, the optimal values for Q are approximately 5 and 7, respectively. Figure 3.4 also demonstrates that using 15 queries is not effective in narrowing the OCI. Similar trends for Q^* are observed with DINOv2 as well, as detailed as shown in Figure 3.5.

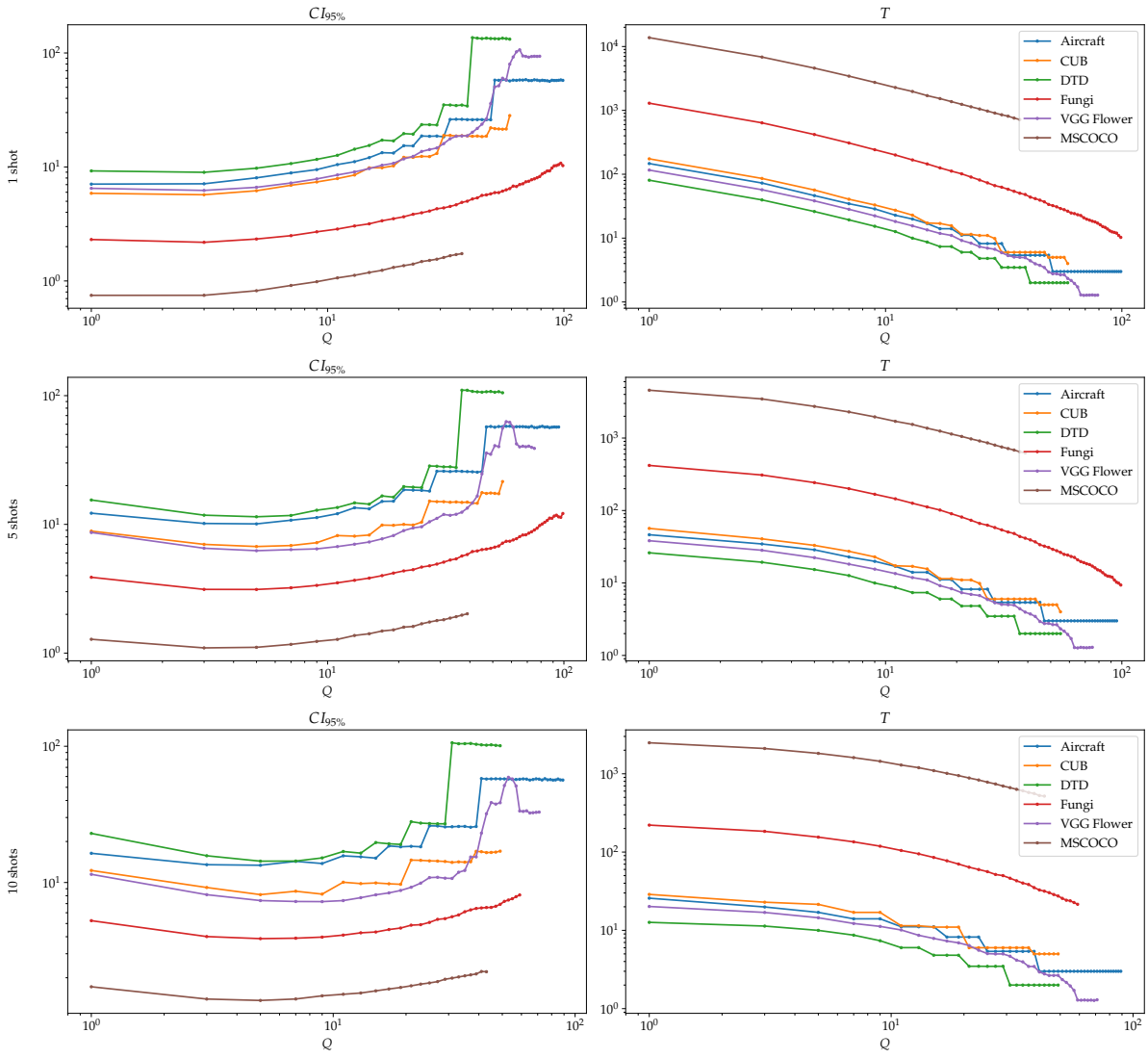


Figure 3.4 – (Left) Confidence Interval ranges (Right) Corresponding number of tasks generated. This experiment is conducted with CLIP. In all graphs the x-axis is the number of queries Q. These results represent averages from multiple trials, with the number of trials tailored according to the Task Count (T). Some curves are stopped before Q reaches 100 because of the number of samples per class. We do not show Omniglot and Quickdraw for visibility. (CC-BY)

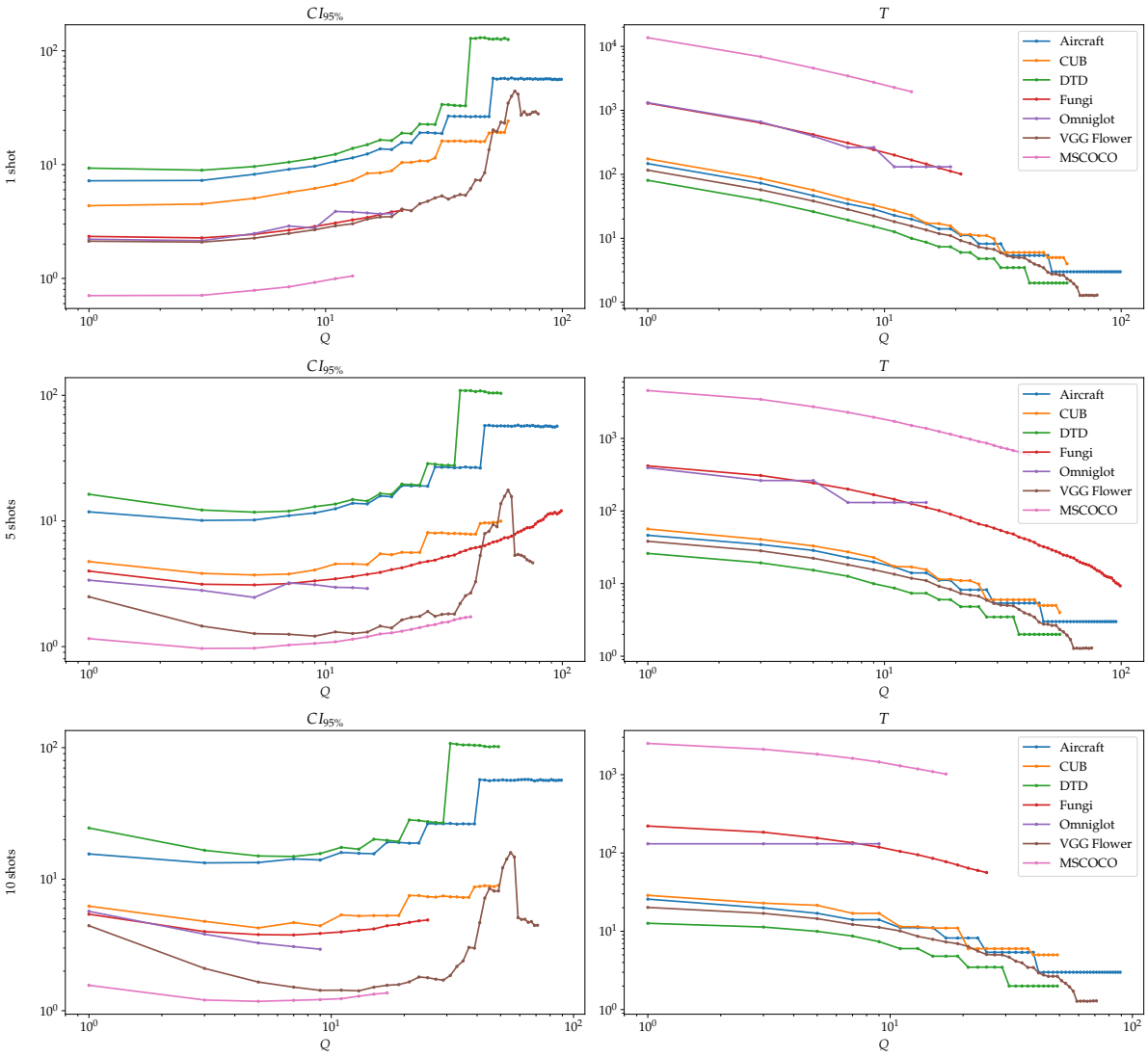


Figure 3.5 – This figure shows that the value of Q^* is not dependent on the model used. This experiment is conducted using DINO v2 instead of CLIP. (Left) Confidence Interval ranges (Right) Corresponding number of tasks generated. In all graphs the x-axis is the number of queries Q. These results represent averages from multiple trials on DINO v2, with the number of trials tailored according to the Task Count (T). Some curves are stopped before Q reaches 100 because of the number of samples per class. We do not show Omniglot and Quickdraw for visibility. (CC-BY)

3.5 Benchmark Proposal

Building on our earlier findings, we proposed [7] a straightforward benchmark that incorporates *Paired Tests* and selects the value of Q as the optimal one identified previously. This approach assumes that the minimum of the confidence interval for Δ also occurs at Q^*, and implicitly that the covariance is independent of Q. These assumptions are supported by the increase in conclusive comparisons shown in Figure 3.6 when optimizing Q and applying Paired Tests. Specifically, while Paired Tests alone resulted in 94 conclusive comparisons, optimizing Q in conjunction with Paired Tests increased this number to 97 conclusive comparisons.

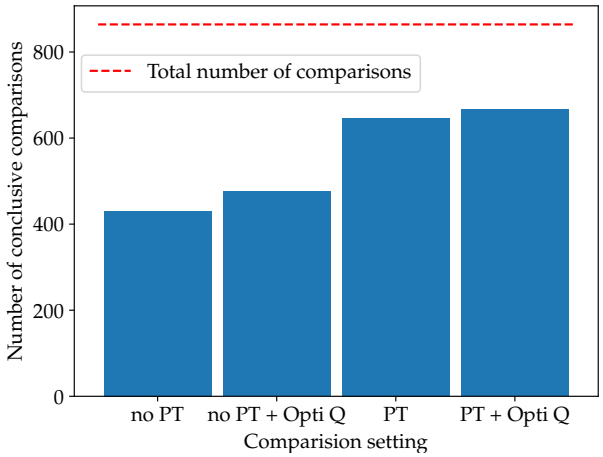


Figure 3.6 – Impact of Paired Tests on the number of conclusive comparisons between all possible pairs formed from combinations of models and methods across datasets (excluding *Omniglot* for technical reasons) for 1, 5, and 10 shots. The Y-axis represents the number of conclusive comparisons. For comparisons where Q is not optimized, we use $Q = 15$. (CC-BY)

This histogram demonstrates that *paired tests* combined with optimized task sizes yield the highest number of conclusive comparisons. The optimization of Q provides a modest improvement over simple paired tests.

Let us define the baselines in this benchmark. In Table 3.4, we present results from our evaluation using DINOv2 as the baseline model and the adaptation methods previously studied. Our experiments consistently reveal that, for a given model, fine-tuning is generally less effective than both logistic regression and nearest class centroid methods. Additionally, the choice of model proves crucial, with DINOv2 showing a distinct advan-

tage over CLIP and DINO across most datasets. Our benchmark, including code, seed values, task descriptions, and accuracy results, is available for reference and use.

	Model	CLIP	DINO	DINOv2	
Dataset	Method	LR	LR	LR	NCM
Aircraft	1-shot	9.241 ± 4.274	25.931 ± 3.693	0.000 ± 1.161	-0.552 ± 1.091
	5-shot	2.286 ± 7.402	16.143 ± 6.311	-2.286 ± 2.043	-1.286 ± 2.027
	10-shot	-0.816 ± 5.803	10.204 ± 5.083	-4.286 ± 2.568	-1.224 ± 2.565
CUB	1-shot	10.743 ± 2.067	25.486 ± 2.991	0.000 ± 0.000	0.000 ± 0.000
	5-shot	2.545 ± 1.410	7.515 ± 2.150	0.121 ± 0.247	-0.121 ± 0.247
	10-shot	1.008 ± 1.464	4.706 ± 3.114	0.000 ± 0.000	0.000 ± 0.000
DTD	1-shot	7.805 ± 4.722	5.122 ± 4.724	0.976 ± 1.941	0.488 ± 1.191
	5-shot	8.750 ± 5.057	3.500 ± 4.109	-0.250 ± 1.979	0.000 ± 1.348
	10-shot	4.762 ± 2.905	1.905 ± 4.392	-3.492 ± 3.253	-1.270 ± 1.937
Fungi	1-shot	15.357 ± 1.262	11.937 ± 1.174	-0.533 ± 0.507	0.235 ± 0.438
	5-shot	10.247 ± 1.369	3.115 ± 1.225	-3.098 ± 0.658	-2.417 ± 0.677
	10-shot	7.316 ± 1.286	0.584 ± 1.224	-2.879 ± 0.890	-1.753 ± 0.799
MSCOCO	1-shot	12.360 ± 1.013	8.820 ± 0.975	0.450 ± 0.377	0.180 ± 0.306
	5-shot	9.952 ± 0.411	6.406 ± 0.389	-0.196 ± 0.187	0.172 ± 0.141
	10-shot	8.181 ± 0.371	3.883 ± 0.354	-1.056 ± 0.204	-0.202 ± 0.158
Omniglot	1-shot	-1.898 ± 1.227	-5.923 ± 1.163	-2.111 ± 0.793	-0.850 ± 0.755
	5-shot	0.760 ± 1.016	-1.932 ± 0.954	-1.308 ± 0.564	-0.973 ± 0.628
	10-shot	0.240 ± 1.136	-2.246 ± 1.027	-1.788 ± 0.669	-0.698 ± 0.733
Quickdraw	1-shot	4.590 ± 1.079	6.530 ± 1.049	-1.170 ± 0.547	1.760 ± 0.526
	5-shot	6.016 ± 0.429	7.850 ± 0.449	-1.028 ± 0.210	-0.160 ± 0.235
	10-shot	5.541 ± 0.331	6.454 ± 0.354	-1.109 ± 0.171	-0.266 ± 0.188
Traffic Signs	1-shot	-1.770 ± 1.055	-0.600 ± 1.034	-0.920 ± 0.519	0.520 ± 0.467
	5-shot	-4.243 ± 0.820	-4.078 ± 0.673	-1.537 ± 0.396	-0.367 ± 0.395
	10-shot	-5.453 ± 0.948	-4.919 ± 0.781	-2.857 ± 0.453	-0.821 ± 0.443
VGG Flower	1-shot	4.576 ± 1.810	12.034 ± 2.243	0.000 ± 0.477	0.000 ± 0.000
	5-shot	0.545 ± 0.829	2.182 ± 1.521	0.000 ± 0.000	-0.182 ± 0.378
	10-shot	0.000 ± 0.000	1.039 ± 0.968	0.260 ± 0.579	0.260 ± 0.579

Table 3.4 – Comparative differences in paired tests. This table contrasts the performance of DINOv2 with Fine-tuning (FT) against DINOv2 combined with Nearest Class Centroid (NCM) or Logistic Regression (LR), as well as the performance combinations of CLIP with DINO using LR. (CC-BY)

3.6 Related Work

Few-Shot Learning Since the landmark studies of [94] and [5], the field of few-shot learning has witnessed significant advancements. Most approaches leverage a pretrained feature extractor, trained on a large, diverse dataset [144, 66, 145]. This feature extractor can either be used directly on the target problem [64] or adapted for improved performance [146]. The variety among proposed methods lies in how they integrate the pretrained feature extractor with their specific adaptation techniques [9]. Initially, benchmarks such as MiniImageNet [94], Omniglot [134], and TieredImageNet [147] focused on *in-domain* scenarios, where the feature extractor is trained on classes that are disjoint from those in the target problem but originate from the same dataset. However, the field has shifted towards *cross-domain* evaluations with the introduction of Meta-Dataset [8] (MD) and later COOP [148], where the feature extractor is trained on a large, generic dataset and then applied to a variety of domains, including fine-grain problems.

Several papers have focused on the concept of sampling tasks with targeted difficulty for few-shot learning. In [149], the authors emphasize that model performance can be enhanced by sampling meta-learning tasks of increasing difficulty. Other works, such as [150] and [151], explore techniques like resampling failed meta-training tasks—identified as challenging—or increasing the likelihood of including previously misclassified samples or classes in future tasks, thereby directing the model's focus to more difficult tasks. Estimating task difficulty remains a complex issue, and various solutions have been proposed to address this challenge [131].

The difficulty-based sampling approach discussed in [131] is pertinent to our research as it allows for the sampling of groups of tasks with uniform difficulty, thereby reducing the range of confidence intervals. However, our study utilizes paired tests, which eliminate the need for such difficulty-based dependencies and offer a more broadly applicable methodology. Paired tests themselves are not a novel aspect of our work. They were originally developed over a century ago to analyze changes in small populations over time [152, 153]. These foundational studies demonstrated that examining individual differences provides more statistical power and insights compared to analyzing average changes across the entire population.

Confidence Intervals The concept of confidence intervals was introduced by Polish mathematician Jerzy Neyman [154] in the early 1930s, complementing Fisher's contemporaneous ideas with a distinct theoretical framework. While confidence intervals gained

broader application and became a standard in medical research by the 1980s, they rely on specific assumptions about the data distribution. In contrast, the bootstrap method, developed by [155], provides a distribution-free approach for estimating ranges. Although our study focuses on traditional confidence intervals due to their widespread use and established understanding in few-shot learning literature, similar insights could be obtained using the bootstrap method.

Challenges in Statistical Interpretation and Methodological Biases The issue of misleading confidence intervals extends beyond our specific field. [156] have highlighted pervasive misinterpretations of confidence intervals across various scientific domains. Compounding this issue is the frequent tendency to overlook or underreport negative or null (non-conclusive) results, which further skews interpretations. [157] emphasizes the need to recognize and thoroughly analyze negative results in computer vision research. Additionally, the impact of dataset biases on evaluation metrics has been extensively documented by [158]. Their study, "Unbiased Look at Dataset Biases," reveals and quantifies various biases, including selection, capture, and negative set biases, underscoring the critical differences between dataset-based findings and real-world performance.

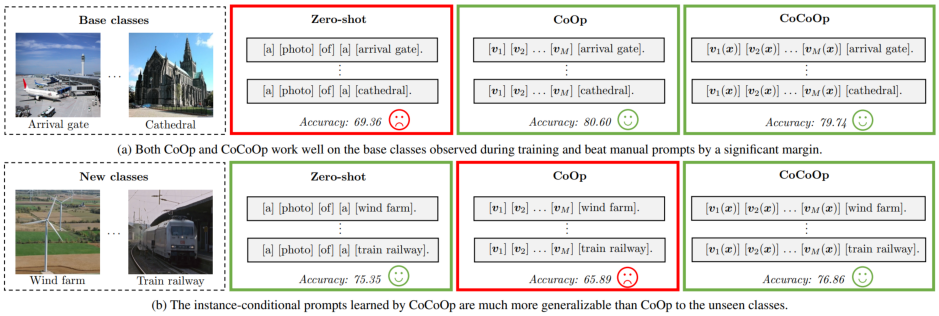
3.7 Limitations

One key limitation of our study is that for large datasets like MSCOCO or QuickDraw, the conventional method of calculating confidence intervals (CI) can result in intervals larger than our proposed OCIs. Therefore, in these cases, using the conventional CI may not be an unreasonable approximation.

Additionally, while our mathematical model explains the origin of the minimum CI in relation to Q, it falls short of providing a method to find this minimum analytically. This is because estimating the parameters α, β, and γ in Equation 3.9 is not straightforward.

Moreover, as discussed at the conclusion of paragraph 3.2.2, achieving 100% accuracy can adversely affect the computation of confidence intervals, particularly impacting the value of the paired test CI in Equation 3.5.

Lastly, we acknowledge that paired tests introduce additional complexity. They require a fixed seed and necessitate saving and sharing individual task accuracies when using the benchmark and comparing methods.



(a) Both CoOp and CoCoOp work well on the base classes observed during training and beat manual prompts by a significant margin.

(b) The instance-conditional prompts learned by CoCoOp are much more generalizable than CoOp to the unseen classes.

Figure 3.7 – The instance-conditional prompts learned by CoCoOp are much more generalizable than CoOp to the unseen classes © [2022] IEEE. Reprinted, with permission, from [Kaiyang Zhou et al., Conditional Prompt Learning for Vision-Language Models , 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 27 September 2022]

3.8 Conclusion

In this study, we highlighted the significant disparity between Open and Closed measurements of method accuracy in Few-Shot Learning. Specifically, while OCIs account for data randomness, they tend to be much wider than CCIs. We pinpointed two key strategies for reducing the width of OCIs and subsequently developed a benchmark incorporating these strategies. Our results emphasize the critical importance of using confidence intervals that reflect data variability in evaluations. We advocate for extending this practice beyond classification and computer vision to all fields that utilize task-based few-shot learning assessments.

3.9 What would we do differently now?

As mentioned in the related work section 3.6, FSL is now increasingly focused on the CoOP [148] benchmark. In this benchmark, we make use of vision-language models and their ability to cope open-set visual concepts. Instead of prompt-engineering a prefix that corresponds to the task “This picture represent a <class> weather”, CoOP relies on the learning prefix prompt tokens as shown in Figure 3.7. Later [159] made the prompt conditional on the input image.

This benchmark uses the evaluation protocol in the CLIP paper [58] with 1, 2, 4, 8 or 16 shots for training. The reported accuracy is averaged over three tasks. The shots are sampled from a train set. The query set is fixed and corresponds to the full test set.

All the available classes are selected in the three tasks ($K = C$). This fixed setting has several key advantages and drawbacks.

Advantages

— The absence of randomness in the shot sampling and query sampling makes the benchmark highly consistent. Its interpretation is less misleading in the sense that the number obtained has no CI.

— All classes within one dataset are sampled which is more realistic.

— Thanks to the visual-language models, the 1, 2, 4, 8 or 16 shots settings can be compared to the zero-shot regime (only using semantic information).

Drawbacks

— CIs are usually not reported since there is no randomness in the task sampling. Thus no conclusion on the distribution can be drawn.

— Averaging on only three tasks might not be enough to fairly assess the performance of a method.

— The fixed query set means tasks are not independent.

— The sampled shots are with replacement meaning that they are even less independent (although the probability of collision for one-shot is low it is not the case for 16 shots). [1]

We think it would be of interest to evaluate how CIs could be created to estimate the accuracy of methods on the distribution from which these datasets are sampled. That would require the division of the dataset between independant splits for each task/seed [2]. It would also be interesting to evaluate the stability of the conclusions if the shots were sampled from other seeds. Is three enough?

1. https://github.com/KaiyangZhou/CoOp/blob/ff61507c790454bce7c5052c3ac39e60772f1f89/lpclip/linear_probe.py at line 29 and 37; **replace = False** simply means that the shots must be different within one class/task but not across tasks.

2. In this setting we consider a new seed for each task see line 29 in the code.

Model	Model Method Dataset	CLIP LR	 NCM	 FT	DINO LR	 NCM	 FT
CLIP	Aircraft	00	00		++	++	++
	CUB	00	00		++	++	++
	DTD	0–	00		0–	0–	00
	Fungi	0–	0–		––	––	––
	MSCOCO	––	0–		––	––	––
	Omniglot	0+	00		0–	––	––
	Quickdraw	0+	00		++	++	++
	Traffic Signs	00	00		0+	++	0+
	VGG Flower	00	00		0+	0+	0+
DINO	Aircraft	––	––	––	0–	00	
	CUB	––	––	––	0–	00	
	DTD	00	00	00	00	00	
	Fungi	++	++	++	0–	0–	
	MSCOCO	++	++	++	0–	0–	
	Omniglot	++	++	++	0+	00	
	Quickdraw	––	––	––	0–	0–	
	Traffic Signs	0–	0–	0–	0–	0+	
	VGG Flower	0–	0–	0–	00	00	

Table 3.3 – Impact of Paired Testing on Significance Relative to Simple Comparison. In this analysis, we compare the Finetune (FT) method against all other methods across both CLIP and DINO models. The initial character in each pair denotes the significance outcome determined without replacement, while the subsequent character reflects the result derived with paired testing. Here, 0 represents a non-significant difference, + indicates a significantly higher accuracy of the FT method, and – conveys a significantly lower accuracy. The FT columns comparing CLIP and DINO are opposites of each other. (CC-BY)

Chapter 4

Less is More: How focusing a model on less data can improve downstream Few-Shot Accuracy?

Contents

4.1 Introduction 96
4.2 Background and related work 97
4.3 Feature extractors for fewer base classes 100
4.3.1 Formulation 100
4.3.2 Choosing class subsets: Informed settings 102
4.3.3 Choosing class subsets: Uninformed setting 103
4.3.4 Heuristics for selecting a feature extractor 104
4.4 Experiments 107
4.4.1 Effect of informed class selection 107
4.4.2 Uninformed setting 112
4.4.3 Implementation details 114
4.4.4 Discussion 115
4.4.5 Ablation study on the number of selected classes 115
4.5 Conclusion 117
4.6 Appendix 118
4.6.1 Impact of learning rate on fine-tuning (DI selection) 118
4.6.2 About batch normalization during fine-tuning 118
4.6.3 A closer look at the unsupervised selection of classes 118
4.6.4 Logistic Regression Few-Shot Classifier 119
4.6.5 Training from Scratch 120
4.6.6 Silhouette scores 120

4.6.7 Segmentation Tasks 122
4.6.8 Feature space distortion or better features? 122
4.6.9 Support set fine-tuning 124
4.7 What would we do differently now? 124

4.1 Introduction

The content of this chapter is largely based on our preprint *Few and Fewer: Learning Better from Few Examples Using Fewer Base Classes* [10] from 2023, later extended and improved in 2024.

As mentioned in the introduction, FSL considers problems where training data is severely limited. It represents a challenge for deep learning, which typically requires large datasets of training examples [144]. In these situations, the standard approach to leverage deep learning involves transfer learning: training or selecting a pre-trained model a model on a large, distinct "base dataset" to create a feature extractor, which then incorporates additional knowledge from the "target dataset" to tackle the specific task. One of the most straightforward transfer strategies is thus to embed the target data into an appropriate feature space, and then to learn a simple classifier with minimal parameters in order to avoid overfitting to the few labeled examples [64].

However, the success of transfer learning is heavily influenced by the similarity between the base and target domains. Recent research [160] indicates that a significant domain gap can actually hinder performance [161]. This chapter tackles the critical question: *Can the domain gap be reduced by fine-tuning on a subset of base classes that closely resemble the target distribution?* By focusing the model's learning on a more relevant and narrowly defined subset of base classes that are closely aligned with the target distribution, this approach seeks to effectively minimize the domain gap and improve transfer learning outcomes.

This challenges the notion that universal feature extractors can consistently deliver *high* performance across *any* few-shot task, a prevalent assumption in the field [162]. While the prevailing literature suggests that leveraging foundational models trained on vast Internet-scale datasets is the optimal approach for new problems with limited data, our approach demonstrates that customized models can significantly outperform generic ones on specific tasks. This finding underscores the No Free Lunch theorem [163], which asserts that no single model is universally superior for all problems.

In this chapter, we explore a powerful concept: starting with an off-the-shelf model trained on a base dataset—referred to as the "base model" or "feature extractor", we propose [10] fine-tuning it using *only* the most relevant classes from that same base dataset. This approach aims to minimize the influence of classes that might negatively impact performance on the target task, while preserving a sufficiently large pool of training examples to avoid overfitting during fine-tuning. Our work prioritizes fine-tuning over pre-training from scratch due to computational constraints, though we also explore pre-training on subsets of the base dataset.

We explore the challenge of selecting a subset of base classes from a dataset, aiming to fine-tune the feature extractor in a way that results in a feature representation with improved inductive bias for a few-shot learning task.

In our study, we work with eight target domains from Meta-Dataset [8] within a cross-domain framework. We find that, for the majority of these domains, fine-tuning a feature extractor with a selected subset of ImageNet base classes enhances the target features used by a Nearest Class Mean (NCM) classifier. Subsequently, we assess our approach under three distinct scenarios: *Domain-Informed* (DI), *Task-Informed* (TI), and *Uninformed* (UI), each representing varying levels of prior knowledge available about the target task.

The primary contributions of this work, previously submitted in our preprint [10], are as follows:

— We demonstrated that accuracy can be improved by fine-tuning with a carefully chosen subset of base classes.

— We proposed straightforward techniques for selecting this subset based on different levels of information about the few-shot task, including either the few-shot examples themselves or unlabelled data from the target domain.

— We explored the potential of using a pre-built library of feature extractors, each fine-tuned on different class subsets. We evaluated various strategies for predefining these class subsets and several heuristics for dynamically selecting a relevant class subset during inference.

4.2 Background and related work

Terminology.

A few-shot classification task, also known as an episode, consists of a support set used to train the classifier and a query set used to test it. The support set includes a limited

number of examples per class. If there are K classes and each class has S examples, the task is referred to as "S-shot K-way" classification.

When benchmarking few-shot learning methods, accuracy is evaluated on the query set and averaged across a large number of tasks. Depending on the context, one may choose between inductive few-shot learning, where each query is classified independently, and transductive few-shot learning, where all queries are processed simultaneously, allowing the classifier to leverage information from their joint distribution. In this chapter, we concentrate on inductive few-shot learning, though the techniques discussed can potentially be adapted for transductive scenarios.

Few-shot paradigms.

In the literature, a common strategy for tackling few-shot tasks is to use a feature extractor pre-trained on a large, general-purpose dataset referred to as the "base dataset." As mentioned earlier, this base dataset is often an internet-scale collection, and pre-trained models are frequently off-the-shelf foundation models. However, to maintain better control over the data used in training, we chose to pre-train our own model instead of relying on foundation models. This approach also ensures a fair comparison with most methods published at the time of our contribution, which adhere to the MetaDataset benchmark, requiring strict control over the base dataset.

Various strategies have been proposed for training efficient feature extractors, including meta-learning methods [6] and closed-form learners [5, 164, 165]. Some approaches focus on directly learning a mapping from support examples and a query input to a prediction [94, 166, 167, 168, 169, 170]. However, straightforward classical batch learning of feature extractors has also demonstrated State-Of-The-Art performance [4], which is why we opt for these simpler feature extractors in our work.

After selecting a feature extractor, various adaptation strategies have been explored [64, 8]. Simple classifiers like Nearest Neighbor or **Nearest Class Mean (NCM)** that require no additional learning [64, 171, 5] have demonstrated competitive performance. Due to their simplicity and effectiveness, we adopt this approach. Recent findings [9] also suggest that our proposed methodology could enhance the performance of any feature extractor training algorithm, providing further motivation for its use.

Lightweight adaptation of feature extractors.

Several previous works have aimed to develop task-specific feature extractors for few-shot learning by introducing a limited number of task-specific parameters into the model, often in the form of residual adapters [172] or Feature-wise Linear Modulation (FiLM)

layers [173]. In a multi-domain context, these parameters can be trained independently for each domain [174, 175]. Alternatively, in other settings, task-specific parameters must either be trained on the support set [176] or predicted from the support set through meta-learning [177, 178, 168]. While feature adaptation has proven effective for multi-domain few-shot learning, it becomes challenging in cross-domain scenarios due to the requirement of training on the support set. Instead, this chapter, based on contributions form our preprint [10], proposes updating *all* parameters of the feature extractor by revisiting the base dataset and fine-tuning on a subset of relevant classes.

Selecting feature extractors or class subsets.

In our work, we consider a setting which requires selecting amongst feature extractors that were each fine-tuned on a subset of base classes. Doing this requires predicting the downstream performance of a feature extractor, a problem already considered in [179]. In this work, they proposed the RankMe metric that is a smooth measure of the feature matrix rank.

[180] proposed to measure task similarity using the Fisher Information Matrix (FIM), and demonstrated the ability of their proposed metric to select a feature extractor trained on an appropriate subset of classes. The experimental section will show that straightforward measures such as cross-validation error perform at least as well as these more involved measures when using a simple classifier in the few-shot setting.

[174] proposed to use a linear combination of features from several domain-specific feature extractors, with coefficients optimized on the support set. Given that our objective is not to obtain the highest accuracy but rather to investigate whether accuracy can be improved using *fewer* base classes, we do not consider mixtures of features in this chapter.

Re-using the base dataset in transfer learning.

For small datasets training, it is critical to avoid over-fitting. As such, several works have explored regularization techniques such as selective parameter updates [181] or auxiliary losses [182, 183, 184, 185]. In contrast, our approach leverages a subset of the base dataset, informed by knowledge of the downstream task. Although this general concept is not entirely new outside the context of few-shot learning, as previous studies have explored using the base dataset for more than task-agnostic pre-training, our application was novel in this domain [10]. For instance, [186] demonstrated that transfer learning can be enhanced by retaining a subset of base dataset classes during fine-tuning, employing separate classifiers and losses for each dataset. Beyond manual class selection, they proposed generating class subsets by solving Unbalanced Optimal Transport (UOT) for the

distance between class centroids in feature space.

Previous studies have used low-level image distances to select a subset of examples (rather than entire classes) for inclusion in fine-tuning [187], or alternatively, selected a subset of classes during the pre-training phase before fine-tuning exclusively on the target dataset [188].

Although the works that focus on selecting class subsets are the most closely related to this chapter, they all rely on fine-tuning with the target set and do not address the few-shot setting, typically using at least 600 examples (which corresponds to 20% of the Caltech 101 dataset).

In contrast, this work focuses on few-shot learning, where fine-tuning on the support set is challenging [9]. We opt to work with subsets of classes rather than individual examples, as this simplifies the problem, helps maintain dataset balance, and offers a straightforward approach for selecting a feature extractor.

Domain adaptation.

In our work, we also introduce a Domain-Informed (DI) setting, which shares some similarities with domain adaptation methods [189, 190] by leveraging unsupervised data from the target domain.

4.3 Feature extractors for fewer base classes

4.3.1 Formulation

Our simple few-shot pipeline comprises three stages:

Step 1. Train a feature extractor with parameters θ with a labeled base dataset containing classes $\mathcal{C}$;

Step 2. Fine-tune the feature extractor on a specific subset of base classes $\mathcal{C}' \subset \mathcal{C}$ to produce the parameters θ';

Step 3. Extract features for the query and support sets and perform NCM classification.

As mentioned in the introduction, the feature extractor is a deep neural network $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$, trained alongside an affine output layer h, such that the composition $h \circ f_\theta$ minimizes the softmax cross-entropy loss [4]. The Nearest Class Mean (NCM) classifier computes the centroid of each class in the feature space and classifies new examples based on the minimum Euclidean distance to these centroids.

The canonical approach to few-shot learning [64] skips directly from Step 1 to Step 3, effectively setting $\theta' = \theta$. However, this places high demands on the feature extractor, expecting it to be *universal*, with its representations immediately applicable to any downstream task, even those from different domains. While much of the prior work in few-shot learning has focused on enhancing Step 3, we propose the hypothesis that fine-tuning on *fewer* base classes (resulting in a smaller dataset) could actually *improve* accuracy in few-shot tasks. A 2D visualization of this effect on a 3-way task is provided below.

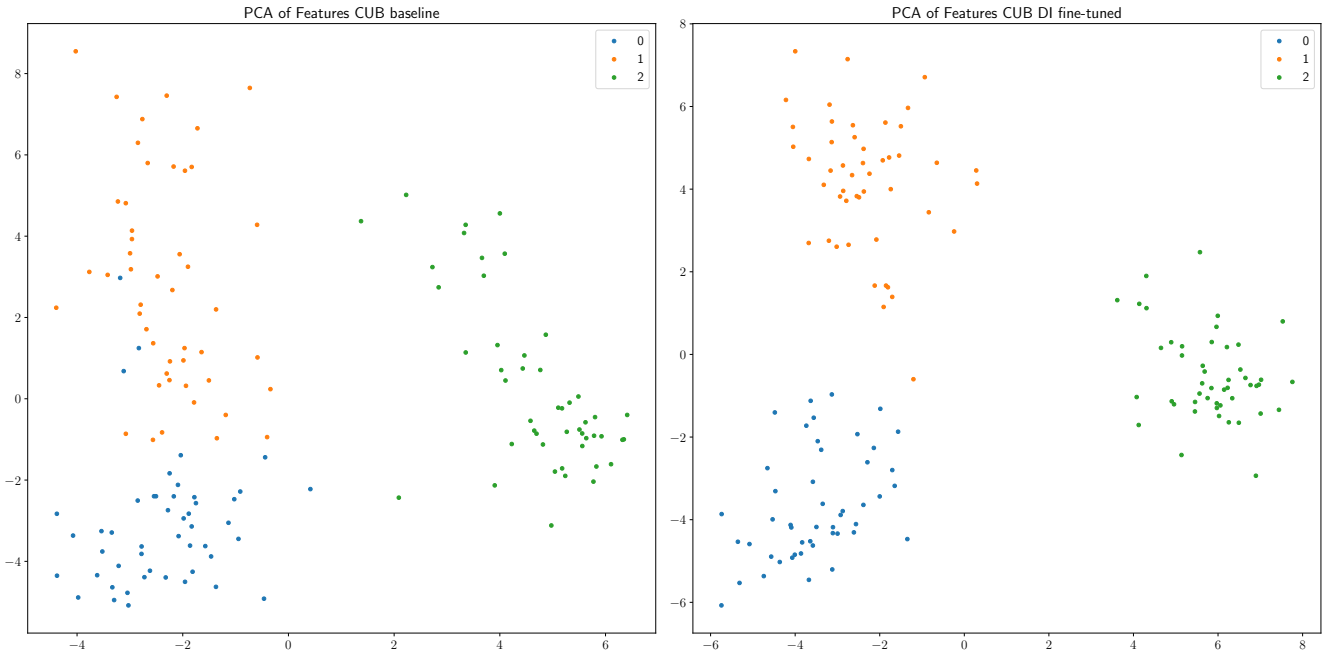


Figure 4.1 – PCA of features of three classes before and after fine-tuning using DI (successful example). We clearly observe an increased separability between classes orange and blue. (CC-BY)

We now address the challenge of selecting a suitable class subset $\mathcal{C}'$. We explore three different settings for class subset selection, each defined by varying levels of task knowledge and constraints on running time. In the **Task Informed (TI)** setting, the support set $\mathcal{S}$ is directly used to select the class subset $\mathcal{C}'$. This represents the ideal scenario, but the computational cost of fine-tuning on a subset of the base dataset may be impractical, especially when multiple few-shot tasks need to be solved or when rapid classifier deployment is required.

The **Domain Informed (DI)** setting considers cases where fine-tuning a feature extractor for each few-shot task is infeasible, but a dataset $\mathcal{D}$—a superset of classes from

the same domain as $\mathcal{S}$—is available to guide class subset selection, without requiring labels. This is a realistic scenario, such as when a robot explores an environment and collects a large number of unlabeled images from the target domain. DI selection also benefits from providing a lower-variance estimate of the class subset, especially as the number of shots in the support set decreases, though it introduces higher bias since the examples do not perfectly match the few-shot task.

Lastly, **Uninformed (UI)** selection defines multiple class subsets $\mathcal{C}_1, \ldots, \mathcal{C}_L$ in advance, without any knowledge of the target domain. This approach introduces the additional challenge of selecting the most appropriate class subset (and corresponding feature extractor) for a given support set. UI is particularly relevant for applications with stringent computational or latency constraints, where a general-purpose set of specialists is required.

The primary baselines to consider are the canonical approach using an NCM classifier (i.e., skipping Step 2 as mentioned earlier) and **fine-tuning on the support set (S)**. The remainder of this section will focus on developing methods for selecting class subsets in each of the previously outlined settings.

4.3.2 Choosing class subsets: Informed settings

The informed settings (TI, DI) involve selecting a subset of base classes $\mathcal{C}' \subset \mathcal{C}$ based on a set of examples $\mathcal{X} = \{x_i\}_i$. In the TI setting, $\mathcal{X}$ corresponds to the support set, whereas in the DI setting, $\mathcal{X}$ refers to the domain examples $\mathcal{D}$, disregarding the labels. The selected class subset $\mathcal{C}'$ will then be used to fine-tune the "base" feature extractor, which was originally trained on the full base dataset.

Algorithm 3 Average Activation selection (TI, DI) (CC-BY)

Require: Base classes $\mathcal{C}$, examples $\mathcal{X} = \{x_i\}$, pre-trained model with feature extractor h and classifier g, class subset size $M = 50$

1: Compute average scores

2: $p = \frac{1}{|\mathcal{X}|} \sum_{x_i \in \mathcal{X}} \text{softmax}(g(h(x_i))$

3: Sort p in descending order **return** $\mathcal{C}' :=$ First M classes of p

To select a class subset, we need a method to identify which base classes are most relevant for a given set of examples $\mathcal{X}$, representing either the task or the domain. Fortunately, the base model already includes a classifier that assigns a score to each base class. We

propose a simple approach: compute the average class likelihoods predicted by the base model on the novel set $\mathcal{X}$, and select the M highest-scoring classes. This straightforward strategy, referred to as **Average Activations (AA)**, is described in Algorithm 3. [1]

While this procedure does not guarantee the selection of the class subset that will yield the optimal representation after fine-tuning, it serves as a low-cost and reasonable approximation. In all experiments, we use $M = 50$ to ensure that the subset sizes are comparable to those in the UI setting, which is described in the following section.

As a point of comparison, we also consider a more advanced selection strategy that requires labels for the set of examples $\mathcal{X}$ used to inform the selection. Specifically, we adopt the Unbalanced Optimal Transport (UOT) formulation from [186], which assigns unit mass to the classes in both $\mathcal{X}$ and $\mathcal{C}$ and uses the distance between class centroids to construct the cost matrix. All regularization parameters follow the settings from [186], and similarly, we select the top $M = 50$ base classes based on the resulting (unnormalized) marginals on $\mathcal{C}$.

4.3.3 Choosing class subsets: Uninformed setting

In the uninformed setting, we address situations where fine-tuning the model on demand is impossible. Our goal is to use *off-the-shelf* techniques to build a *static library* of specialized feature extractors, each trained on class subsets determined in an unsupervised manner. This allows us to select an appropriate class subset based on the support set without additional fine-tuning.

To achieve this, we apply agglomerative hierarchical clustering to the base classes using Ward's method [191]. Each class is represented either by its centroid under the base feature extractor h_θ (visual features, V) or by a vector embedding of its name from the text encoder of the publicly available CLIP model [58] (semantic features, Se). As recommended in the original CLIP paper [58], we prepend "a photo of a" to the class labels to improve representation quality.

We obtain final clusters by setting a distance threshold that produces eleven relatively balanced clusters for the 712 classes in the ImageNet training split of Meta-Dataset [8]. This process is repeated for the concatenation of visual and semantic features (denoted X), with both types of feature vectors normalized and centered prior to concatenation. As a baseline for comparison, we also create a random (R) partitioning of the base classes

1. Although the softmax cross-entropy loss is invariant to an additive constant, this does not affect the ranking of the unnormalized logits.

into eleven subsets.

After clustering, a separate feature extractor is fine-tuned for each class subset, resulting in a static library of class subsets and corresponding model parameters $(\mathcal{C}'_j, \theta'_j)$. The base model $(\mathcal{C}, \theta)$ is also included in the static library.

4.3.4 Heuristics for selecting a feature extractor

Lastly, we address the challenge of selecting the most appropriate specialist feature extractor given the support set for a new few-shot task. To tackle this, we propose a set of heuristics that are expected to correlate with accuracy on the query set. These heuristics leverage the labeled support set $\mathcal{S}$, the feature extractor $h_{\theta'_j}$, and the class subset $\mathcal{C}'_j$ that was used during fine-tuning.

We describe the heuristics here:

Leave-One-Out (LOO). Leave-One-Out (LOO) is a form of cross-validation applied to the support set. In this approach, validation is performed by randomly selecting a single element from the support set, excluding it from the calculation of class centroids. This process is repeated multiple times, and the results are averaged to estimate accuracy. To speed up computations, we isolate one sample *per class* in our experiments. LOO is essential because it minimizes the impact on the training set, which is crucial in few-shot settings. In such conditions, removing even a small number of elements from the support set can significantly affect the classifier's performance.

Signal-to-Noise Ratio (SNR). Signal-to-Noise Ratio (SNR) is another metric that correlates with accuracy. In the case of isotropic Gaussian distributions, it serves as an ideal heuristic for predicting theoretical accuracy. For two classes i, j, SNR is defined as follows:

$$SNR(i,j) = \frac{\delta}{\xi} = 2\frac{\|\mathbb{E}(\boldsymbol{\mathcal{N}}^i) - \mathbb{E}(\boldsymbol{\mathcal{N}}^j)\|_2}{\sigma(\boldsymbol{\mathcal{N}}^i) + \sigma(\boldsymbol{\mathcal{N}}^j)} \tag{4.1}$$

where $\mathbb{E}(\boldsymbol{\mathcal{N}}^i)$ and $\sigma(\boldsymbol{\mathcal{N}}^i)$ represent the empirical expectation and standard deviation of class i in the support set, respectively. In this context, δ is the margin between the class centroids, and ξ refers to the noise, represented by the sum of the standard deviations. For tasks involving more than two classes, the SNR is computed as the average over all pairs of classes.

Support Set Accuracy (SSA) SSA measures the accuracy of the support set using an NCM (Nearest Class Mean) classifier. In this case, we evaluate a few-shot task where the support set also serves as the query set, meaning the classifier is tested on the same examples used for training. This provides a direct estimation of how well the model fits the support set.

Support Set Confidence (SSC) SSC provides a confidence score and serves as a soft version of SSA. It assesses how tightly grouped the shots are around their respective centroids.

$$SSC \simeq \mathbb{E}\left(\max_{i}\left(\underset{k}{\text{softmax}}\left(\frac{-\mathbf{d}_{i,k}}{T}\right)\right)\right) \tag{4.2}$$

Here, $\mathbf{d_{i,k}}$ represents the distance between a support sample i and the centroids k, with T being the temperature parameter. Essentially, it measures how well the class samples are clustered around their centroids.

Monte-Carlo Sampling (MCS). Monte-Carlo Sampling [192] generates virtual examples by sampling from regularized Gaussian distributions fitted to each class in the support set to create an artificial validation set. In MCS, we first compute an empirical covariance matrix and centroid for each class in the support set within the feature space. Virtual data points are then generated in this space to reflect the distribution of the support set. These virtual data points are classified in the same way as any query examples, and the resulting accuracy is used as a proxy for the actual performance. For single-shot scenarios, an isotropic variance is applied.

Rank-Me (RKM). Rank-Me [179] is a heuristic that correlates with the performance of downstream tasks, as introduced in [179]. This method involves defining a soft version of the rank [193] and measuring this pseudo rank on a feature matrix derived from a model. Higher ranks are indicative of better performance. In this approach, we use the features of the support set to compute the rank.

FIM FIM corresponds to the Fischer Information Matrix as described in [180]. Fisher Information Matrix provides a measure of task similarity using a probe network. We have directly made use of their code to create embeddings for tasks and datasets. The Fisher

information matrix (F) is defined as

$$F = \mathbb{E}_{x,y\sim\hat{p}(x)p_\theta(y|x)}\left[\nabla_\theta \log p_\theta(y|x)\nabla_\theta \log p_\theta(y|x)^T\right] \tag{4.3}$$

that is, the expected covariance of the scores (gradients of the log-likelihood) with respect to the model parameters θ. x are inputs and y are labels. $\hat{p}$ is the empirical distribution defined by the training set and $p = \sigma \circ h \circ f_\theta$ the baseline model. f_θ is the feature extractor, h the classification head and σ the sigmoid function. The distance is measured using the normalized diagonals of the FIM of datasets. Cosine distance is used. We used the probe networks proposed by [180]. We compute the distances between clusters of base classes and support sets from few-shot tasks.

AA Finally, Average Activation (AA), which was previously used for subset selection, can also be applied here. Average Activation selects the cluster of classes that is most activated by the support set of a task. The index of the chosen cluster s is given by:

$$s = \underset{i}{\text{argmax}} \left(\sum_{c\in\mathcal{P}_i; x\in\mathcal{S}} p_\theta(y_c|x) \right) \tag{4.4}$$

where $p_\theta(y_c|x)$ represents the activation of the base class c for a given image x in the support set $\mathcal{S}$. $\mathcal{P}_i$ denotes the cluster of base classes i. We select the class subset that has the highest cumulative activation in the support set for the task.

In contrast to other methods, the SNR, AA, and FIM heuristics can be used with just a single shot. SNR addresses this challenge by focusing solely on between-class covariance in the one-shot scenario. Additionally, except for AA, all heuristics require evaluating the candidate feature extractor. Consequently, selecting a class subset will involve exhaustively evaluating all feature extractors in the library, which typically involves only a few dozen models.

To assess the effectiveness of our heuristics, we compare them against a random heuristic (RH) that selects a feature extractor uniformly at random, as well as an oracle that always chooses the feature extractor with the highest accuracy on the validation set. The oracle represents an upper bound on the best possible performance for a given set of few-shot tasks and feature extractors. However, this level of performance might not be attainable with the available information.

4.4 Experiments

We present results on eight datasets from the Meta-Dataset, excluding ImageNet and QuickDraw. These datasets include Omniglot (handwritten characters), Aircraft, CUB (birds), DTD (textures), Fungi, VGG Flowers, Traffic Signs, and MSCOCO (common objects) [134, 135, 136, 137, 138, 139, 140, 142].

In the following, fine-tuning on the support set is one of our baselines, denoted as S. We use the methodology proposed by [8] for fine-tuning with some minor differences detailed in the Appendix.

In the following analysis, fine-tuning on the support set serves as one of our baselines, denoted as S. We employ the fine-tuning methodology outlined by [8], with some minor modifications detailed in the Appendix.

We consider three sampling procedures for generating few-shot tasks: 1-shot 5-ways, 5-shots 5-ways, and the task-sampling procedure described by Meta-Dataset [8], referred to as MD. The MD procedure involves tasks with a significantly larger but variable number of shots and ways.

We report both the baseline accuracy and the performance improvement, or *boost*, denoted as Δ, with respect to the baseline. For each dataset and sampling procedure, we sample a fixed set of 600 few-shot tasks, which remains constant across all methods (S, TI, DI, DI-UOT, TI-UOT).

Since accuracy is measured using the same set of tasks for all methods, the confidence interval for the accuracy boost can be computed using paired trials. The confidence intervals for the baselines represent the distribution of the sample mean across the 600 tasks.

4.4.1 Effect of informed class selection

The primary objective of our first experiment is to examine how fine-tuning feature extractors on a subset of base classes before applying NCM classification impacts the few-shot accuracy. In this experiment, we focus on the Average Activation (AA) selection strategy in both Task-Informed (TI) and Domain-Informed (DI) settings. We compare these results against fine-tuning directly on the support set, as well as the Unbalanced Optimal Transport (UOT) selection strategy [186] applied in both DI and TI settings.

This comparative analysis allows us to evaluate how these different approaches affect performance in few-shot tasks, shedding light on the effectiveness of various class subset

selection strategies for fine-tuning.

Table 4.1 presents the baseline accuracies and relative performance boosts across all settings for each dataset and few-shot sampling procedure. Table 4.2 shows the results of using DI using logistic regression rather than the NCM classifier, following the no-replacement sampling method described in Chapter 3. While most conclusions remain consistent across the distributions, DTD stands as an exception, where the sample size is too small to draw definitive conclusions. Throughout the paper, except for Table 4.2, all confidence intervals (CI) are standard CCIs as defined in Chapter 3. In our discussion of Table 4.1, which uses CCIs, we note that the results closely align with those from logistic regression and OCIs. To clarify, OCIs refer to Open Confidence Intervals, which are CIs derived when tasks are sampled without replacement. Further results, using CCIs with logistic regression and MD sampling, are provided in Table 4.3 in the Appendix.

The results demonstrate that Domain-Informed (DI) selection of base classes can lead to substantial accuracy improvements. On average, DI selection yields a boost of $+1.62 \pm 0.08$ points across all datasets and sampling methods. However, the performance boost varies significantly across datasets, and this disparity is most pronounced for Traffic Signs and Aircraft. For Traffic Signs, a consistent decline in accuracy is observed, except when fine-tuning is applied with a sufficient number of shots. This is likely due to the lack of visually similar images in ImageNet. For instance, while ImageNet contains around 50 bird classes relevant to CUB, the most strongly activated class for Traffic Signs is *Nematode*, which is unrelated to the domain of traffic signs. This is compounded by the lower resolution and tightly cropped nature of Traffic Sign images, which differ significantly from the higher-resolution images in ImageNet. The feature extractor, trained on high-quality ImageNet images, may struggle to effectively capture fine details in low-resolution images, further hindering its performance on Traffic Signs. It appears across all experimental settings that any fine-tuning is particularly detrimental to the few-shot performance on Traffic Signs.

Similarly, minimal gains on Aircraft are likely due to the limited presence of relevant ImageNet classes, such as *airliner* and *military plane*, which may act as supersets of the few-shot task classes. Relatively poor boosts are obtained on the Aircraft dataset because it is almost fully captured by just two base classes. The 'airliner' class may contribute to a collapse of all Aircraft classes, causing confusion between them and reducing performance. Paradoxically, closely related classes could be detrimental rather than helpful in this case, suggesting that more research is needed to explore the potential impact of class collapse.

Table 4.1 – The table shows the change in performance when fine-tuning on the support set (S), using Task-Informed (TI) subset selection, Domain-Informed (DI) subset selection, and DI-UOT subset selection. Positive boosts with overlapping confidence intervals are highlighted in bold. Overall, DI achieves the best results, followed by TI, while S yields the lowest performance. Additionally, the UOT selection strategy is outperformed by the simpler AA selection. The full table, including UOT results for each dataset, is provided in the appendix. (CC-BY)

Dataset	Method	1-shot 5-ways		5-shots 5-ways		MD	
		Baseline	Δ	Baseline	Δ	Baseline	Δ
Aircraft	S	39.95 ±0.70	-3.60 ±0.64	63.18 ±0.74	-1.48 ±0.61	65.86 ±0.90	**+5.33 ±0.69**
	TI		-0.06 ±0.33		**+0.26 ±0.31**		+1.33 ±0.25
	DI		**+0.34 ±0.32**		**+0.54 ±0.31**		+1.32 ±0.27
CUB	S	64.34 ±0.90	-19.28 ±0.88	87.78 ±0.59	-18.97 ±0.63	79.29 ±0.90	-14.51 ±0.60
	TI		+2.64 ±0.44		**+2.16 ±0.26**		+1.08 ±0.19
	DI		**+3.27 ±0.44**		**+2.29 ±0.26**		**+2.20 ±0.20**
DTD	S	45.21 ±0.77	+0.66 ±0.77	70.10 ±0.59	-3.12 ±0.59	76.03 ±0.69	-6.67 ±0.69
	TI		**+2.85 ±0.46**		**+2.77 ±0.33**		**+2.44 ±0.29**
	DI		**+2.90 ±0.48**		**+2.96 ±0.33**		**+2.78 ±0.31**
Fungi	S	53.01 ±0.92	-6.59 ±0.74	74.87 ±0.80	-8.33 ±0.62	51.57 ±1.16	-15.05 ±0.53
	TI		**+0.92 ±0.39**		**+1.67 ±0.30**		**+1.07 ±0.26**
	DI		**+1.07 ±0.41**		**+1.89 ±0.29**		**+1.38 ±0.25**
Omniglot	S	61.80 ±1.03	-3.16 ±1.11	81.53 ±0.76	**+3.53 ±0.85**	59.51 ±1.31	-4.59 ±1.07
	TI		**+2.65 ±0.38**		**+2.94 ±0.29**		**+3.74 ±0.23**
	DI		**+3.52 ±1.22**		**+3.57 ±0.81**		**+3.93 ±0.61**
MSCOCO	S	43.91 ±0.85	-5.44 ±0.66	63.04 ±0.79	-6.20 ±0.63	44.99 ±0.99	-17.00 ±0.72
	TI		**+1.27 ±0.35**		**+1.87 ±0.29**		+1.85 ±0.17
	DI		**+1.62 ±0.34**		**+2.09 ±0.30**		**+2.25 ±0.17**
Traffic Signs	S	**57.35 ±0.85**	-4.67 ±0.66	74.11 ±0.78	**+6.17 ±0.62**	**53.77 ±1.05**	**+0.77 ±1.00**
	TI		**-0.84 ±0.32**		-1.22 ±0.25		-2.02 ±0.17
	DI		**-0.79 ±0.95**		-1.48 ±0.77		-1.82 ±0.44
VGG Flower	S	75.86 ±0.84	+0.19 ±0.79	94.46 ±0.33	-1.45 ±0.37	92.77 ±0.58	-5.18 ±0.51
	TI		**+2.04 ±0.40**		**+0.64 ±0.18**		**+1.03 ±0.16**
	DI		**+1.88 ±0.41**		**+0.52 ±0.18**		**+0.84 ±0.16**
Average	S		-5.24 ±0.78		-3.73 ±0.61		-7.11 ±0.73
	TI		**+1.43 ±0.38**		**+1.39 ±0.28**		**+1.31 ±0.21**
	DI		**+1.73 ±0.57**		**+1.55 ±0.41**		**+1.61 ±0.30**
	DI-UOT		+0.63 ±0.47		+0.36 ±0.33		+0.32 ±0.28
	TI-UOT		**+1.43 ±0.36**		**+1.10 ±0.44**		**+1.21 ±0.32**

Table 4.2 – The table shows the performance gains using Logistic Regression (5-ways 15-queries) and compare the different sampling method and Confidence Intervals as explained in Chapter 3. Δ is the result of paired tests. (CC-BY)

1-shot	No Replacement			With Replacement		
Dataset	Baseline	DI	Δ	Baseline	DI	Δ
DTD	52.444 ± 5.013	53.481 ± 4.832	1.037 ± 4.303	48.990 ± 1.687	52.638 ± 1.653	3.648 ± 1.061
VGG Flower	80.615 ± 4.754	80.513 ± 4.105	-0.103 ± 1.831	77.600 ± 1.642	79.743 ± 1.602	2.143 ± 0.757
Aircraft	40.157 ± 2.766	42.353 ± 3.365	2.196 ± 1.267	42.124 ± 1.399	42.962 ± 1.550	0.838 ± 0.695
CUB	72.392 ± 3.900	74.745 ± 4.361	2.353 ± 1.418	67.343 ± 1.880	70.657 ± 1.840	3.314 ± 0.861
Omniglot	78.830 ± 1.924	82.422 ± 1.590	3.593 ± 2.369	79.581 ± 1.747	83.610 ± 1.613	4.029 ± 2.099
Fungi	50.735 ± 1.782	52.644 ± 1.817	1.910 ± 0.685	55.056 ± 1.882	56.567 ± 1.950	1.512 ± 0.767
Traffic Signs	57.328 ± 1.014	55.493 ± 0.972	-1.835 ± 0.993	59.543 ± 1.701	58.552 ± 1.840	-0.990 ± 1.758
MSCOCO	42.010 ± 0.510	43.004 ± 0.524	0.994 ± 0.203	44.914 ± 1.643	45.657 ± 1.740	0.743 ± 0.766
5-shots	No Replacement			With Replacement		
Dataset	Baseline	DI	Δ	Baseline	DI	Δ
DTD	73.905 ± 4.285	75.429 ± 3.802	1.524 ± 2.006	73.162 ± 1.260	74.962 ± 1.229	1.800 ± 0.597
VGG Flower	95.556 ± 1.639	95.889 ± 1.643	0.333 ± 0.626	95.267 ± 0.551	95.886 ± 0.490	0.619 ± 0.325
Aircraft	67.429 ± 3.598	65.810 ± 4.095	-1.619 ± 1.851	68.990 ± 1.572	68.810 ± 1.600	-0.181 ± 0.652
CUB	90.917 ± 3.474	92.167 ± 3.028	1.250 ± 1.019	90.019 ± 0.978	91.162 ± 0.987	1.143 ± 0.434
Omniglot	94.280 ± 0.762	96.346 ± 0.527	2.066 ± 0.829	94.457 ± 0.705	96.029 ± 0.564	1.571 ± 0.856
Fungi	74.552 ± 1.811	75.769 ± 1.728	1.217 ± 0.673	76.589 ± 1.422	77.572 ± 1.426	0.983 ± 0.570
Traffic Signs	79.010 ± 1.024	77.787 ± 1.009	-1.223 ± 0.796	81.800 ± 1.328	81.152 ± 1.386	-0.648 ± 1.414
MSCOCO	61.352 ± 0.606	62.246 ± 0.611	0.894 ± 0.200	65.095 ± 1.609	65.905 ± 1.586	0.810 ± 0.644

These factors account for the considerable variability in performance boosts in the DI setting, as further detailed in the Appendix. Figure 5.11 helps to clarify these results in the context of the initial table presented in the chapter.

Interestingly, the assumption that class selection would only result in significant gains for relatively easy tasks, or when the base feature extractor is already quite effective, is not supported by the results. In fact, the improvements tend to be inversely correlated with the baseline accuracy, with greater boosts observed when the initial accuracy is lower. This inverse correlation between accuracy and performance boost is empirically evidenced in Figure 4.4. For example, the poor performance on Aircraft and Traffic Signs may also stem from the fact that these tasks rely more heavily on shape representation, whereas the base feature extractor may prioritize color and texture, which are more beneficial for datasets like CUB, VGG Flower, and DTD. Further exploration of this hypothesis is required.

The results show that the Unbalanced Optimal Transport strategy [186] yields improvements that are either comparable to or worse than the simple Average Activation approach. Notably, there is a significant performance decline on Omniglot, which has the largest number of classes in its test split (659), suggesting that the algorithm's hyperparameters are likely sensitive to the scale of the task and apparenlty led to a poor convergence of the Sinkhorn algorithm used to compute the UOT [194].

The set of classes chosen by the Unbalanced Optimal Transport (UOT) method differs substantially from those selected by the Average Activation (AA) strategy. We observed that the Intersection over Union (IoU) between these class sets varied, ranging from 22% for MSCOCO to 78% for CUB.

Task-Informed selection is frequently observed to slightly underperform compared to Domain-Informed selection. This is particularly evident in the case of CUB, where the base dataset includes a substantial number of relevant classes (birds) that can be leveraged for class subset retrieval. This highlights the higher variance associated with selecting class subsets from fewer examples (as shown in the Appendix). The results suggest that the bias inherent in Domain-Informed selection is more advantageous than the variance of Task-Informed selection, even in settings with larger data availability.

Fine-tuning on the support set (S) can be highly effective, particularly in higher data regimes such as the 5-way 5-shot and Meta-Dataset (MD) task sampling. For instance, boosts of up to $\Delta = +6.17 \pm 0.62$ points were observed for 5-way 5-shot classification on Traffic Signs. The baseline accuracy for Traffic Signs is notably low, likely due to the lack

of relevant data in the base dataset. Consequently, fine-tuning on the support set can have a significant positive impact, where other methods might only amplify or diminish the influence of less relevant classes in the base dataset. A similar effect, albeit on a smaller scale, may also be observed for Aircraft.

During our experiments, we found that fine-tuning on the support set is highly sensitive to hyperparameter choices. Among the various configurations we tested (details provided in the Appendix), fine-tuning often resulted in a considerable decrease in performance. This suggests that the main challenge of this method is identifying the optimal hyperparameters for each task without a validation set.

When the domain is known, Domain-Informed selection proves to be the most reliable method for improving few-shot accuracy. This is particularly true in low-data scenarios such as 1-shot 5-way classification, where it benefits significantly from the information provided by unlabeled examples. Even in mixed sampling scenarios with more shots, DI maintains its advantage, though the difference is less pronounced. When the domain is unknown, Task-Informed selection is generally a safer alternative compared to fine-tuning on the support set, which can sometimes lead to poor outcomes.

Overall, the results demonstrate that training with a smaller subset of base classes can substantially enhance accuracy compared to using the full feature extractor. This supports the notion that fine-tuning with a class subset can improve performance. Additionally, we assessed this increased accuracy by measuring the silhouette score [195]. The silhouette score for target features improved by $\Delta = +0.0103$ across all datasets, compared to an average baseline silhouette score of -0.001.

4.4.2 Uninformed setting

Our second main experiment focuses on the Uninformed (UI) setting. We aim to assess whether it is possible to achieve a performance boost relative to the baseline without prior knowledge of the task during fine-tuning. This involves comparing methods for unsupervised class subset construction and feature extractor selection. The results are illustrated in Figure 4.2, which shows the performance improvements for each domain and selection heuristic using MD sampling with both concatenated (X) and random (R) subset constructions.

First, we note that we achieved a significant boost in accuracy in most cases. Both MCS and SSA consistently improved performance across all our experiments when used with the X subset design. This is particularly noteworthy as it highlights the potential

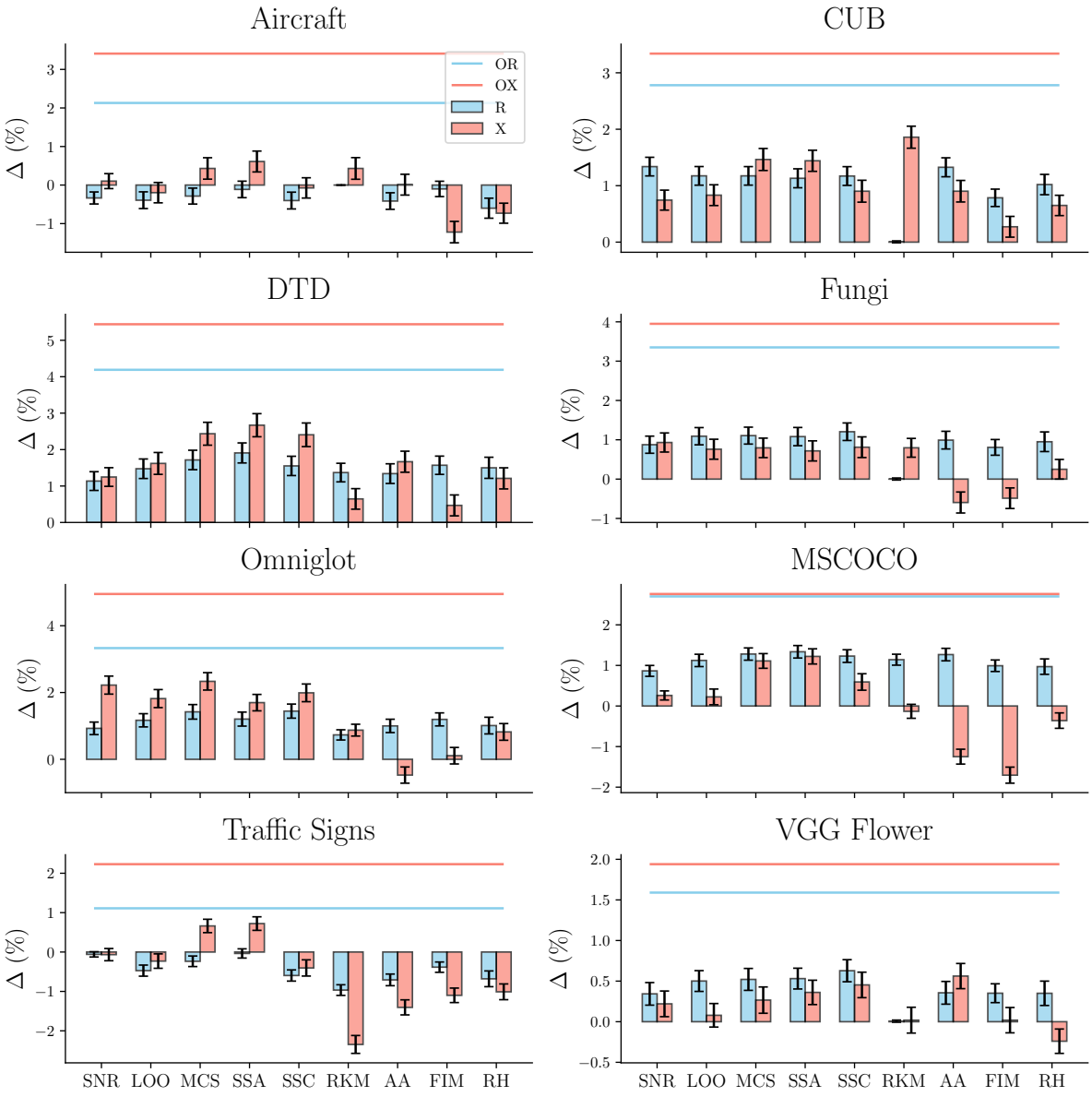


Figure 4.2 – Difference in accuracy compared to the baseline after selecting feature extractors using various heuristics. Tasks are sampled according to the MD protocol. In R (resp. X), heuristics select a feature extractor from the R (resp. X) library of feature extractors. The oracle OR (resp. OX) identifies the best feature extractor for each task within the R (resp. X) library. The Random Heuristic (RH) chooses a feature extractor at random. SSA and MCS are the top-performing heuristics. A well-chosen class (X) is particularly beneficial for datasets showing high performance boosts. (CC-BY)

for deploying such solutions in scenarios with stringent computational and latency constraints, supporting our second claim from the introduction.

It is not surprising that X generally outperforms R, especially in datasets where im-

provements are substantial, indicating that a well-designed subset is preferable. The X-based oracle often achieves notably higher accuracy than its R-based counterparts although it is not the case on certain datasets like MSCOCO. Some heuristics, such as AA and FIM, seem to perform poorly with X. This issue arises especially with MSCOCO, a dataset closely related to the ImageNet distribution. This suggests that meaningful subset construction is particularly crucial when the target dataset is finer-grained or less similar to the base dataset. Results for V, Se. (detailed in Figure 5.10 in the Appendix), and X are comparable, with a slight edge for V, especially on the Traffic Signs dataset. Nonetheless, we chose to present results for X due to its combination of orthogonal cues, making it more robust in novel domains. We also show results with similar conclusions in the 1-shot and 5-shot regimes in Figures 5.9 and 5.8.

Among the different heuristics, Support Set Accuracy (SSA) performs best on average across datasets and subset constructions under MD sampling, with an average boost of $\Delta = +1.13 \pm 0.22$ points. For 5-shot 5-way tasks, Monte-Carlo Sampling (MCS) delivers the highest boost with $\Delta = +0.78 \pm 0.27$ points. In 1-shot 5-way tasks, the Signal to Noise Ratio (SNR) heuristic provides the best boost with $\Delta = +0.74 \pm 0.38$ points. Notably, even under the challenging condition of a single shot per class, significant accuracy improvements are achievable by using a feature extractor fine-tuned on a pre-determined subset of base classes. The large gap between our methods and the oracle (denoted by O) suggests that the maximum achievable boost is consistently above 2%, with potential gains reaching up to 6%. Our heuristic outperforms previous methods such as FIM [180] and RKM [179]. However, the heuristic based on Average Activation of the base classifier was found to be less reliable across domains compared to our baseline.

4.4.3 Implementation details

In both the TI and S settings, fine-tuning is conducted for each task, limiting our ability to explore the hyperparameter space for every case. This constraint is particularly significant in the TI setting, where a full two-step fine-tuning process involving 50 classes is required for each task, dataset, and sampling configuration. In contrast, for the DI setting, we leverage the validation split to select class subsets, ensuring the process is task-independent but still domain-dependent.

We used the Adam optimizer [196] to train the classifier during the first step and SGD with Nesterov momentum of 0.9 [197] for the complete fine-tuning in the second step. The learning rate was set to 0.001 with a cosine scheduler [105] to ensure comparability

across settings. To focus on the effect of data selection, we limited the dataset size to 10k examples. Fine-tuning was conducted for 10 epochs in the first step (with a frozen feature extractor) and for 20 epochs in the second step (with an unfrozen feature extractor). A simple ResNet-12 architecture was used as the feature extractor.

In the Appendix, we demonstrate that the DI setting can be further improved by using heuristics to select feature extractors fine-tuned with different learning rates. We followed standard procedures for training f_θ, as outlined in [164, 4]. Experiments were run on two GPU clusters, utilizing Nvidia A100s and V100s, while an Nvidia 3090-equipped machine was used for prototyping our methods.

4.4.4 Discussion

We have also demonstrated that our findings extend to segmentation tasks, as shown in Table 4.6 in the appendix. Our work touches on a broad range of questions, many of which remain unexplored. For instance, we only briefly address the geometric and ontological relationships between source and target classes, which are likely crucial in predicting the direction and magnitude of the accuracy boost. Additionally, self-supervised fine-tuning may outperform other approaches in scenarios where (a) none of the base classes are directly relevant to the target domain, or (b) ontological relationships, such as inclusion, exist between base and target classes (e.g., using a base class like "dog" to classify between different breeds).

In the UI setting, we fixed the number of clusters, and in the DI and TI settings, we fixed the number of selected classes. Our analysis demonstrates that this choice is near-optimal for most datasets. Furthermore, a heuristic for determining the appropriate subset size is provided in Figure 4.3. Future research could delve into analyzing these methods in the context of domain shifts between support and query examples [198]. While we intentionally left these areas unexplored to focus on other aspects, they present promising directions for future investigation.

4.4.5 Ablation study on the number of selected classes

Another limitation of our study is the high computational cost associated with certain heuristics (FIM, MCS, and LOO) and settings (TI, TI-UOT, and to a lesser extent, S). As mentioned earlier, fine-tuning on the support set can be highly beneficial, but its success is often hindered by challenges in optimizing hyperparameters. We believe that methods

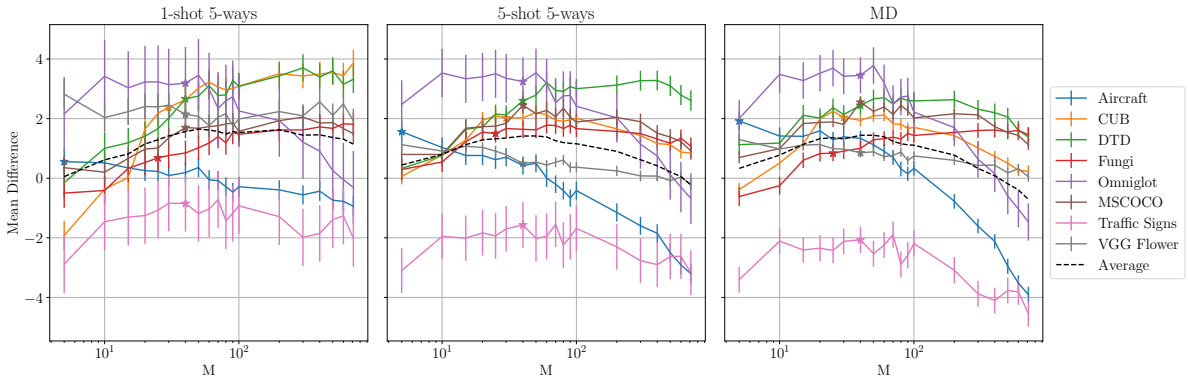


Figure 4.3 – Relative gain in accuracy compared to the baseline after fine-tuning (Domain Informed setting), with varying numbers of classes M selected using the Average Activation (AA) method. The star markers indicate points where 90% of the cumulative activation across classes is achieved. For all datasets except Aircraft and Fungi, the 90% cumulative activation threshold is reached at approximately the same number of classes ($M \sim 40$), which corresponds to the peak relative gain compared to the baseline. Figure 5.11 illustrates the distribution of activation across classes. (CC-BY)

capable of predicting the accuracy of few-shot tasks could be invaluable for setting these parameters more effectively.

Finally, we believe that fine-tuning is not the only solution for adapting embeddings to a task. Carefully designed, data-dependent projections could offer fast, "on-the-fly" solutions to boost performance, especially in situations where computational constraints or task-specific adaptations are necessary.

Finally, can our results generalize to other models? According to [9], training and adaptation algorithms are largely uncorrelated. This suggests that our adaptation method could be effectively applied to any pre-training algorithm, offering flexibility across different model architectures.

However, it is important to note that our approach focuses exclusively on inductive classification. While inductive methods are frequently emphasized in academic research, they are often less relevant in real-world industrial applications. In practice, tasks such as predicting few-shot accuracy and using query set information to improve embeddings are more critical for optimizing performance in these environments.

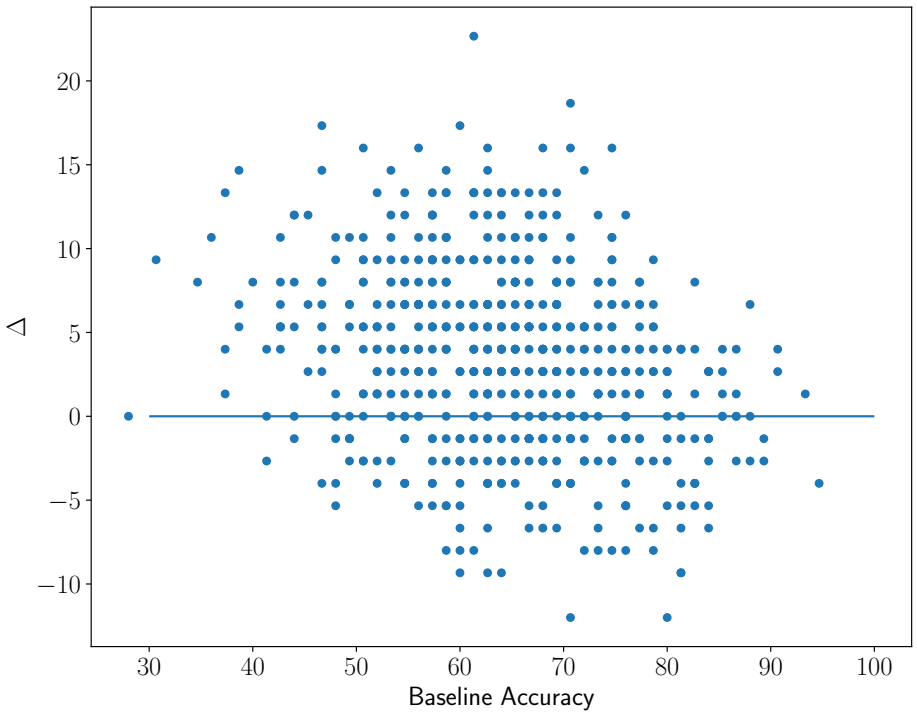


Figure 4.4 – DI Boost versus baseline accuracy in 1-shot 5-ways for CUB. This refutes the hypothesis that only problems which already enjoy high accuracy can benefit from subset selection: rather, there is a negative correlation between boost in accuracy and baseline accuracy (as mentioned in the chapter). The regular grid stems from the discrete set of possible outcomes for 75 query examples (5 ways with 15 query examples per class). (CC-BY)

4.5 Conclusion

In conclusion, this chapter presented several approaches for identifying relevant subsets of base classes that, when fine-tuned, can significantly enhance accuracy in few-shot tasks. Notably, fine-tuning on a subset selected from the unlabeled target domain proved to be the most consistent strategy for improving performance. However, this approach does not yield the same benefits across all datasets, leaving many questions unanswered. We hope this will encourage the community to further explore this effect, particularly the impact of dataset scale. Additionally, we proposed a straightforward strategy for building an offline static library of feature extractors, from which one can be dynamically selected

for few-shot tasks. As interest in foundational models that serve as universal embeddings for downstream tasks continues to grow, we believe our work provides an intriguing counterpoint for future research.

4.6 Appendix

4.6.1 Impact of learning rate on fine-tuning (DI selection)

We fine-tuned the baseline model on DI-selected subsets using various learning rates, as illustrated in Figure 5.4, which demonstrates that the improvement in accuracy over the baseline is highly sensitive to the chosen learning rate.

Given the significant impact learning rate has on final accuracy, we further propose using our heuristics to identify the most appropriate learning rate. It's important to note that AA and FIM cannot be used to select learning rates for feature extractors, as these methods are based solely on data selection, independent of the model. In Figures 5.5 and 5.6, we observe substantial performance improvements across most datasets. These figures indicate that, for certain datasets, the heuristically selected learning rates can outperform the previously reported rate of 0.001 from the main chapter.

4.6.2 About batch normalization during fine-tuning

During the fine-tuning process, both the feature extractor parameters and batch normalization statistics can be adjusted to fit the selected subset. The most recent statistics are estimated using moving averages, while the parameters are updated through gradient descent. To explore this further, we conducted experiments where the learning rate was set to 0, meaning only the batch normalization statistics were updated. As shown in Figure 5.4, updating only the batch normalization statistics can significantly enhance performance, particularly in the case of Omniglot.

4.6.3 A closer look at the unsupervised selection of classes

The dendrograms constructed using Ward's method are shown below. In Figures 5.13 and 5.14, we zoom in on the bird cluster. Both semantic and visual features display a remarkable ability to identify a bird-related cluster, though each also includes some anomalies (non-birds) and omits certain birds. Further details are provided in the figure

captions. This highlights the reasoning behind our approach, which leverages both visual and semantic features (X) for selection.

4.6.4 Logistic Regression Few-Shot Classifier

We confirmed that our method enhances feature quality even when using a Logistic Regression (LR) classifier instead of the NCM classifier. As shown in Table 4.3, DI fine-tuning consistently improves the representations, demonstrating a positive impact across both classifiers.

Table 4.3 – Accuracy obtained using a Logistic Regression (LR) classifier for both the baseline and proposed methodology (instead of NCM). (CC-BY)

Dataset	1-shot 5-ways		5-shot 5-ways		MD	
	Baseline	Δ	Baseline	Δ	Baseline	Δ
Aircraft	42.28 ±0.72	+0.07 ±0.42	68.50 ±0.69	-0.08 ±0.33	72.47 ±0.99	+0.88 ±0.27
CUB	65.28 ±0.82	+4.81 ±0.47	86.74 ±0.58	+2.91 ±0.29	77.85 ±0.93	+2.81 ±0.21
DTD	50.57 ±0.78	+1.88 ±0.50	72.02 ±0.60	+2.15 ±0.37	80.21 ±0.74	+1.92 ±0.31
Fungi	55.58 ±0.90	+0.45 ±0.45	76.32 ±0.76	+1.47 ±0.33	42.78 ±1.09	+2.59 ±0.27
Omniglot	66.41 ±0.99	+2.77 ±1.14	87.68 ±0.63	+1.86 ±0.66	64.45 ±1.35	+3.33 ±0.61
MSCOCO	46.97 ±0.87	+0.83 ±0.41	66.57 ±0.73	+0.60 ±0.32	44.42 ±1.09	+1.38 ±0.18
Traffic Signs	62.36 ±0.85	-1.16 ±0.93	83.96 ±0.64	-1.18 ±0.61	59.73 ±1.12	+2.78 ±0.38
VGG Flower	80.11 ±0.72	+1.50 ±0.38	95.15 ±0.30	+0.63 ±0.19	91.19 ±0.61	+1.91 ±0.19
Average	58.69 ±0.44	+1.39 ±0.23	79.62 ±0.35	+1.04 ±0.15	66.64 ±0.58	+2.20 ±0.12

4.6.5 Training from Scratch

We compared the DI finetuning to training from scratch on the same DI subsets. We report the results in Table 4.4. We see that only for omniglot, training on fewer, more similar classes, helps. We used the same hyperparameters to train our baseline model.

This suggests that refining a universal feature extractor may be more effective than training directly on a specialized dataset. This approach aligns seamlessly with the paradigm of foundation models.

Table 4.4 – Accuracy obtained when deploying the proposed methodology training from scratch on DI subsets (instead of finetuning). (CC-BY)

Dataset	1-shot 5-ways		5-shot 5-ways		MD	
	Baseline	Δ	Baseline	Δ	Baseline	Δ
Aircraft	39.95 ±0.70	-7.77 ±0.62	63.18 ±0.74	-19.12 ±0.64	65.87 ±0.90	-25.28 ±0.60
CUB	64.34 ±0.90	-8.13 ±0.85	87.78 ±0.59	-9.58 ±0.53	79.29 ±0.90	-14.43 ±0.42
DTD	45.21 ±0.77	-1.24 ±0.76	70.10 ±0.60	-6.98 ±0.53	76.03 ±0.69	-8.83 ±0.53
Fungi	53.01 ±0.92	-11.25 ±0.78	74.87 ±0.79	-15.54 ±0.61	51.57 ±1.16	-15.87 ±0.50
Omniglot	61.80 ±1.03	+3.10 ±1.26	81.53 ±0.76	+2.85 ±0.84	59.51 ±1.31	+3.82 ±0.66
MSCOCO	43.91 ±0.85	-5.52 ±0.62	63.04 ±0.79	-9.39 ±0.58	44.99 ±0.99	-10.37 ±0.35
Traffic Signs	57.35 ±0.85	-5.17 ±1.00	74.11 ±0.78	-4.17 ±0.77	53.77 ±1.05	-5.03 ±0.46
VGG Flower	75.86 ±0.84	-8.80 ±0.82	94.46 ±0.33	-6.94 ±0.46	92.77 ±0.58	-8.63 ±0.40
Average	55.18 ±0.44	-5.60 ±0.33	76.13 ±0.38	-8.61 ±0.29	65.47 ±0.55	-10.58 ±0.29

4.6.6 Silhouette scores

In Table 4.5, we present silhouette scores [195] (calculated using scikit-learn) that demonstrate improved separability of target classes due to DI fine-tuning. These scores offer insight into how well each sample is positioned within its class, reflecting both the compactness and separation of the classes. Across all datasets, the silhouette score for features in different classes increased by $\Delta = +0.0103$, compared to an average baseline silhouette score of -0.001.

Dataset	**Baseline**	**Ours**
Aircraft	0.0104	0.0096
CUB	0.0303	0.0317
DTD	0.0293	0.0460
Fungi	-0.0436	-0.0342
Omniglot	-0.0719	-0.0016
Traffic Signs	-0.0279	-0.0374
VGG Flower	0.0919	0.0913
MSCOCO	-0.0277	-0.0224
Average	-0.0011	0.0103

Table 4.5 – Comparative Analysis of Silhouette Scores for Features Extracted Using Two Different Backbones Across Diverse Datasets. (CC-BY)

4.6.7 Segmentation Tasks

Table 4.6 shows that our DI finetuning improved representations for segmentation tasks as well. This highlights the ability of our method to improve performances on different types of tasks.

Table 4.6 – mIOU, mIOU reduced (hard classes ignored, specific to Cityscape) and accuracy on the segmentation dataset of Cityscape [199] using the method developed in [200]. Our experiments compare DI feature extractors with our baseline feature extractor on the same seeds (paired tests). (CC-BY)

Metric	1-shot 5-ways		5-shot 5-ways	
	Baseline	DI	Baseline	DI
mIOU	18.46 ± 0.26	18.72 ± 0.25	22.76 ± 0.13	23.07 ± 0.13
mIOU Reduced	21.87 ± 0.31	22.17 ± 0.30	26.92 ± 0.15	27.28 ± 0.15
Accuracy	70.49 ± 0.38	71.13 ± 0.33	74.24 ± 0.14	74.78 ± 0.13

4.6.8 Feature space distortion or better features?

To determine whether the observed improvement is due to mere distortion rather than an enhancement in representation, we conducted experiments in which the backbone was kept fixed and only an additional linear layer (with bias) was trained on the class subset. This setup allows the linear layer to potentially distort the feature space without fundamentally altering the representation.

The results, presented in Table 4.7, indicate that training only this linear layer leads to a decrease in the accuracy of the NCM classifier. This finding suggests that fine-tuning does indeed improve the representation rather than merely distorting the feature space.

Table 4.7 – Difference in performance between using the features from the backbone directly vs. adding an extra linear layer, using DI subsets. We use the NCM classifier on top each time. The Mode column give more details about the extra layer that is placed just before the classification head. This extra layer is trained with the classification head in the same way as step 1. When "Finetune" is added to the mode we simply also apply step 2 (unfrozen backbone). "640x640" corresponds to a randomly initialized linear layer. "640x640 Res" corresponds to the same layer initialized at 0 with a skip connection (c.f. ResNet architecture). (CC-BY)

Dataset	Mode	1-shot 5-ways	5-shot 5-ways	MD
Average	640x640 Finetuned Res	$+1.71 \pm 0.35$	-0.08 ± 0.25	-1.19 ± 0.20
	640x640 Res	$+0.59 \pm 0.35$	-2.18 ± 0.24	-3.66 ± 0.20
	640x50	-1.69 ± 0.37	-6.64 ± 0.28	-10.44 ± 0.26
	Finetuned 640x50	-0.80 ± 0.38	-5.60 ± 0.28	-8.83 ± 0.25
	640x640	-0.32 ± 0.37	-3.77 ± 0.26	-5.82 ± 0.23
Aircraft	640x640 Finetuned Res	-1.11 ± 0.85	-5.16 ± 0.76	-7.81 ± 0.59
	640x640 Res	-1.10 ± 0.84	-7.10 ± 0.73	-10.85 ± 0.61
	640x50	-5.05 ± 0.86	-14.11 ± 0.74	-21.84 ± 0.66
	Finetuned 640x50	-4.50 ± 0.81	-13.76 ± 0.77	-20.64 ± 0.66
	640x640	-2.40 ± 0.87	-10.51 ± 0.73	-15.48 ± 0.60
CUB	640x640 Finetuned Res	$+8.02 \pm 0.93$	$+1.82 \pm 0.48$	$+0.49 \pm 0.41$
	640x640 Res	$+9.37 \pm 0.93$	$+1.24 \pm 0.48$	-0.41 ± 0.46
	640x50	$+6.37 \pm 0.97$	-1.80 ± 0.50	-6.38 ± 0.52
	Finetuned 640x50	$+7.86 \pm 0.94$	-0.92 ± 0.48	-4.82 ± 0.50
	640x640	$+9.13 \pm 0.96$	$+0.51 \pm 0.49$	-1.73 ± 0.47
DTD	640x640 Finetuned Res	$+3.79 \pm 0.99$	$+1.65 \pm 0.66$	-0.48 ± 0.63
	640x640 Res	$+2.02 \pm 0.97$	$+0.62 \pm 0.66$	-2.05 ± 0.61
	640x50	$+3.91 \pm 1.01$	-2.16 ± 0.63	-7.06 ± 0.67
	Finetuned 640x50	$+5.63 \pm 0.99$	-2.02 ± 0.66	-6.73 ± 0.65
	640x640	$+2.74 \pm 1.02$	$+0.11 \pm 0.65$	-3.07 ± 0.62
Fungi	640x640 Finetuned Res	-0.54 ± 0.99	-3.09 ± 0.67	-2.82 ± 0.56
	640x640 Res	-1.72 ± 0.98	-5.14 ± 0.67	-4.60 ± 0.56
	640x50	-2.53 ± 0.98	-8.89 ± 0.69	-10.24 ± 0.60
	Finetuned 640x50	-2.60 ± 0.98	-8.25 ± 0.67	-8.03 ± 0.58
	640x640	-1.71 ± 0.99	-6.29 ± 0.67	-6.16 ± 0.62
MSCOCO	640x640 Finetuned Res	$+1.20 \pm 0.98$	$+4.04 \pm 0.70$	$+3.05 \pm 0.46$
	640x640 Res	$+1.81 \pm 0.99$	$+1.81 \pm 0.75$	$+0.59 \pm 0.45$
	640x50	$+1.99 \pm 1.02$	$+0.52 \pm 0.76$	-3.13 ± 0.46
	Finetuned 640x50	$+2.44 \pm 0.97$	$+1.08 \pm 0.76$	-1.94 ± 0.46
	640x640	$+1.80 \pm 0.98$	$+1.17 \pm 0.77$	-0.81 ± 0.45
Omniglot	640x640 Finetuned Res	-0.42 ± 1.19	-0.84 ± 0.84	-1.61 ± 0.57
	640x640 Res	-5.16 ± 0.88	-6.10 ± 0.55	-7.43 ± 0.49
	640x50	-11.48 ± 0.95	-15.09 ± 0.63	-17.28 ± 0.61
	Finetuned 640x50	-9.04 ± 1.27	-11.78 ± 0.93	-13.66 ± 0.66
	640x640	-7.93 ± 0.95	-9.24 ± 0.58	-10.78 ± 0.50
Traffic Signs	640x640 Finetuned Res	$+2.06 \pm 0.92$	$+2.14 \pm 0.71$	$+1.19 \pm 0.44$
	640x640 Res	$+1.02 \pm 0.95$	-0.00 ± 0.70	-0.95 ± 0.43
	640x50	-1.85 ± 0.94	-4.81 ± 0.74	-8.44 ± 0.45
	Finetuned 640x50	-1.86 ± 0.95	-3.12 ± 0.73	-7.01 ± 0.46
	640x640	-0.06 ± 0.94	-0.87 ± 0.72	-2.32 ± 0.43
VGG Flower	640x640 Finetuned Res	$+0.65 \pm 0.91$	-1.17 ± 0.38	-1.54 ± 0.33
	640x640 Res	$+0.31 \pm 0.95$	-2.75 ± 0.38	-3.57 ± 0.34
	640x50	-4.83 ± 0.96	-6.81 ± 0.47	-9.19 ± 0.43
	Finetuned 640x50	-4.35 ± 0.93	-5.99 ± 0.47	-7.81 ± 0.41
	640x640	-4.16 ± 1.01	-5.06 ± 0.43	-6.19 ± 0.40

4.6.9 Support set fine-tuning

We adhered closely to the protocol outlined in [8] for fine-tuning on the support set, employing several configurations. We reproduced three of these configurations, which are reported as the best results. The performance, as shown in Table 4.8, is generally quite low. This suggests that overfitting may be a significant issue due to the small size of the training dataset.

4.7 What would we do differently now?

As noted in the introduction, this work commenced in 2022 and concluded in 2023, reflecting the FSL paradigm of that period.

To align with current practices, we would adapt this work to the contemporary paradigm by applying the two-step adaptation process to foundation models using subsets of their training datasets.

These foundation models are frequently trained using contrastive learning [58] or self-supervised learning [132]. It remains uncertain whether the improvements observed in this study would transfer to these models. Our subset adaptation could take several forms:

— Classical supervised learning [201]
— Supervised contrastive learning
— Self-supervised methods

Since foundation models are trained on internet-scale datasets, applying Average Activation (AA) on such large-scale data poses significant challenges. Consequently, extensive efforts would be required to obtain representative prototypes of subdomains in the training data.

Table 4.8 – Performance of fine-tuning on the support set with varying hyperparameters. We could not explore more than three settings as these require long computational effort. All Positive values are highlighted. Frozen signifies that only the last classification layer was trained while the rest of the network was frozen. (CC-BY)

Sampling	Dataset	Frozen; $lr = 10^{-3}$	$lr = 10^{-3}$	$lr = 10^{-4}$
1-shot 5-ways	Aircraft	-12.24 ±0.73	-3.60 ±0.64	-6.64 ±0.66
	CUB	-18.04 ±0.86	-19.28 ±0.88	-25.52 ±0.97
	DTD	-3.74 ±0.88	**0.66 ±0.77**	-6.32 ±0.78
	Fungi	-10.90 ±0.81	-6.59 ±0.74	-14.91 ±0.87
	Omniglot	-29.13 ±1.06	-3.16 ±1.11	-21.17 ±1.08
	MSCOCO	-4.47 ±0.70	-5.44 ±0.66	-9.09 ±0.70
	Traffic Signs	-8.37 ±0.90	-4.67 ±0.66	-8.73 ±0.75
	VGG Flower	-20.69 ±1.15	**0.19 ±0.79**	-16.78 ±0.95
	Average	-13.45 ±0.85	-5.24 ±0.75	-13.64 ±0.81
5-shot 5-ways	Aircraft	-24.55 ±0.85	-1.48 ±0.61	-11.71 ±0.67
	CUB	-15.60 ±0.82	-18.97 ±0.63	-25.50 ±0.70
	DTD	-16.33 ±0.84	-3.12 ±0.59	-9.49 ±0.62
	Fungi	-13.86 ±0.81	-8.33 ±0.62	-18.28 ±0.76
	Omniglot	-39.31 ±1.08	**3.53 ±0.85**	-22.57 ±1.09
	MSCOCO	-5.11 ±0.66	-6.20 ±0.63	-11.43 ±0.65
	Traffic Signs	-4.03 ±0.81	**6.17 ±0.62**	-0.67 ±0.62
	VGG Flower	-17.36 ±0.96	-1.45 ±0.37	-9.87 ±0.57
	Average	-17.02 ±0.81	-3.73 ±0.59	-13.69 ±0.68
MD	Aircraft	-33.49 ±0.90	**5.33 ±0.69**	-16.98 ±0.82
	CUB	-18.49 ±0.65	-14.51 ±0.60	-39.36 ±0.80
	DTD	-24.93 ±0.94	-6.67 ±0.68	-11.06 ±0.63
	Fungi	-18.90 ±0.65	-15.05 ±0.53	-30.75 ±0.66
	Omniglot	-40.25 ±1.02	-4.59 ±1.07	-36.27 ±1.01
	MSCOCO	-8.85 ±0.44	-17.00 ±0.72	-20.21 ±0.50
	Traffic Signs	-14.70 ±0.57	**0.77 ±1.00**	-16.93 ±0.65
	VGG flower	-34.71 ±1.05	-5.18 ±0.51	-25.93 ±1.02
	Average	-24.29 ±0.76	-7.11 ±0.71	-24.69 ±0.74

Chapter 5

General Conclusion

5.1 Summary of contributions

In this thesis, we have explored several strategies to enhance the robustness, fairness, and adaptability of Few-Shot Learning (FSL) models. Our work addresses key challenges in FSL, focusing on building universal models that generalize well across tasks, establishing more reliable evaluation protocols, and optimizing task-specific adaptation through a data-centric approach.

In the second chapter, we presented a straightforward feature extractor designed for few-shot classification in both inductive and transductive settings. By integrating it with augmented samples and ensembling techniques, we demonstrated its capability to achieve state-of-the-art accuracy when paired with simple classifiers across multiple standardized benchmarks. In several instances, our approach surpassed previous methods by a notable margin, exceeding 1% improvement in accuracy.

In Chapter 3, we demonstrated that the confidence intervals commonly reported in the FSL literature can be misleading, as the conclusions drawn from them are often specific to particular datasets rather than being representative of the overall data distribution. We presented two methods to obtain more conclusive comparisons between pairs of FSL methods. These include paired tests and task sizing.

In chapter 4, we showed that particularization to a FSL task from a general feature extractor can be done in two steps. The gap is progressively closed using by finetuning on a related subset of the pre-training or *base* dataset, then by adapting to the task in a traditional way. This, to our knowledge is the first demonstration of improvement in the FSL setting.

5.2 General answer to the Problem statement

As demonstrated in Chapter 4, some components of the pre-training dataset might be forgotten or discarded from the model weights to enhance transfer learning for few-shot tasks. Consequently, we address the issue of adversarial samples in the training set of the feature extractor. Our pre-adaptation approach allows us to focus more easily on selecting which data to retain rather than which to discard. Additionally, we assessed the optimal size of the subset of classes to retain in comparison to the full dataset. This optimal size appears to be significantly influenced by the relationship between the base and target domains.

This finding lends support to research focused on data-centric AI [202] and core set approaches [203]. While the introduction of this thesis suggested that more data generally improves performance, our results add nuance to this view. A single high-performance general-purpose model may not be the ultimate solution for every problem. This underscores the No Free Lunch theorem [163], which asserts that no single model is universally optimal for all tasks. We hope this insight will also stimulate research into the adversarial nature of specific modules, neurons, or feature dimensions within models for particular tasks.

5.3 Outlooks and Future works

5.3.1 What is a real-world scenario?

In a context where companies are looking to integrate AI into their operations, researchers may question the practical relevance of their focus. Criticizing and upgrading academic benchmarks is not new [198, 204, 8, 205, 206]. In a large-scale study spanning health institutions, scientific research, and industry, a comprehensive survey and steering committee of their specific few-shot learning needs would provide valuable insights. Understanding the fundamental mismatches between computer science academia and outside setting could foster the development of benchmarks that more closely reflects the complexities of real-world problems.

Several key questions about what constitutes a real-world scenario remain open. A first one within the context of computer vision is: how often is classification sufficient as opposed to using more informative methods such object detection [207] or segmentation [208]?

In the following, we limit the scope of the proposal to classification as it is the main focus of this thesis. Even within classification several questions remain: for instance, what target accuracy levels are necessary for successful deployment in real applications? Indeed, once the target accuracy is reached it might be less relevant to obtain further gains of performance. Should the support and query sets be sampled from the same distribution, as highlighted by [198]? Should the support set be balanced, or can it be skewed to reflect real-world class imbalances, as alluded to in [8]?

Moreover, many real-world scenarios involve the presence of data that does not belong to any predefined class, underscoring the importance of Few-Shot Out-of-Distribution Detection (FSOOD) [209]. In FSOOD, the challenge extends beyond classifying known categories; it requires models to identify and correctly handle unknown or undefined classes that should be categorized as *other*. Addressing this challenge enhances the robustness and practicality of few-shot learning models when confronted with diverse, unpredictable data distributions.

Furthermore, in certain real-world contexts, there may be a greater emphasis on maximizing recall for specific critical classes, potentially at the expense of precision. Investigating these trade-offs, along with the broader questions of class imbalance and distributional shifts, would pave the way for research more closely aligned with solving real-world problems.

To address these questions effectively, we propose a two-step evaluation process inspired by ELO scores [210], designed to assess both the relevance of tasks and the effectiveness/relevance of methods in few-shot learning.

Step 1: Task Evaluation Initially, institutions would use a dedicated software platform featuring a range of few-shot learning tasks, without associated methods. The platform would present pairs of tasks to the participants repeatedly. For each pair, participants would evaluate and rank the tasks based on their relevance to real-world applications, including factors such as:

— Practical Applicability: How well the tasks represent real-world challenges.
— Complexity and Diversity: The breadth of scenarios covered by the tasks.
— 0 and few-shot mix: Some scenarios might include classes defined by images and/or text and/or sound and/or other modalities. For example, some rare animals may only have been heard, their sound could be indicative for an detection using images.
— Other features of a task that were not listed here.

This process helps identify which tasks are most relevant and useful for practical applications, providing a clearer understanding of what types of problems are most pressing for industries and researchers.

Step 2: Method Evaluation In the second phase, the focus shifts to evaluating various few-shot learning methods. Using the previously selected tasks, the software platform would now present pairs of methods applied to these tasks. Participants would rank the methods based on criteria such as:

- Accuracy and Robustness: Performance metrics including precision, recall, and handling of out-of-distribution samples.
- Class Imbalance Handling: Effectiveness in managing skewed class distributions. (if relevant on the task)
- Explanability and Interpretability
- Other information that might be relevant.

By separating the task selection from the method evaluation, this approach ensures that the methods are assessed based on the most relevant and challenging tasks identified in the first step. This two-tiered evaluation process helps industries and researchers understand both the types of tasks they should focus on and which methods are most effective for those tasks. It also provides valuable insights into the current literature, guiding future research and practical implementations.

Finally, because the software is open-source, users can load their own problems locally. This allows them to choose whether to disclose their data and to evaluate and rank methods based on their specific tasks.

5.3.2 Feature Space Dynamics in transfer learning: Onthological relations bewteen source and target

Ontological relationships between source (base) and target classes were discussed in the thesis as potential factors that could either enhance or hinder performance of transfer learning, though no definitive conclusions were reached. This project proposal aims to explore this aspect more thoroughly.

For example, if the pre-training dataset includes broad classes (e.g., "dog") that serve as supersets for some or all of the target classes (e.g., "malamute"), does this negatively impact performance? Conversely, what happens if the target classes are supersets of the

base classes? This question is particularly important, as it is often assumed that training samples from the same class tend to converge toward a single point in feature space, which may reduce the ability to later differentiate between subclasses within that broader class.

In self-supervised and contrastive learning settings, the structure of the feature space differs significantly [211, 212, 213]. How does this affect the conclusions previously drawn?

5.3.3 Alternative Methods to Incorporate Semantic Information in the FSL-VLM Framework

In the current framework for few-shot learning (FSL), Vision Language Models (VLMs) offer several advantages:

Clarifying tasks: FSL tasks are often subject to multiple interpretations. For instance, the task may focus on the background rather than the central object in an image. Providing a semantic representation of class labels can help mitigate this ambiguity [61].

Reducing shot variance: A semantic class representation can inform the model of a class's semantic center, helping to minimize shot variance. The shot might deviate from this center for valid reasons—such as the targeted class not fully matching the semantic description—or for less valid reasons, like the shot being a poor class prototype. This raises an important question. In [133], it is demonstrated that for certain datasets, learning with a linear probe on up to four shots does not surpass zero-shot performance. This prompts the inquiry: when do shots provide real value, and how should visual and semantic information be effectively combined?

However, utilizing semantic information in FSL is not limited to this approach. Could such information guide the adaptation of visual models? A study has shown that CLIP reveals property-specific roles of many attention heads (e.g., location or shape) [214]. We suggest that coupling class semantic information with structured reweighting or pruning of foundation models could be a promising approach. If the task involves birds, could the contributions of attention heads unrelated to birds be reduced or even eliminated? This pruning strategy could provide a hardware-efficient method to compress and specialize models without traditional or parameter-efficient fine-tuning (e.g on embedded platforms without backpropagation). This method might offer a more precise correction than simply adding semantic information in VLMs. Moreover, it could potentially extend to purely visual models, assuming their modules can be semantically interpreted.

5.3.4 The Future Role of LLMs in FSL

Vision-Language Models (VLMs) are already capable of handling language-intensive tasks like Visual Question Answering (VQA) [215] and Image Captioning [216]. However, their language understanding and reasoning capabilities remain limited compared to pure LLMs. What can external, dedicated LLMs contribute to enhance these models?

For practitioners with proprietary data (e.g., in sectors such as industry, defense, or healthcare), LLMs could provide significant value by guiding models with information not encountered during the pretraining of foundation models. For instance, users could prompt the model to avoid specific pitfalls and direct it toward relevant aspects of the task. As an example, a geologist might indicate that certain features in the data are irrelevant to the task while others are potentially important, though their relevance is uncertain. This opens the door for Human-in-the-loop AI, enabling highly specialized applications.

Additionally, LLMs can enhance the output phase by offering detailed explanations for the model's decisions. They could generate a comprehensive list of potential classification patterns, along with corresponding rationales, and invite users to choose the most appropriate interpretation for their specific needs.

This prompts important questions: What role might LLMs play in the future of vision-based few-shot learning (FSL)? What unique knowledge or capabilities could they offer that VLMs alone cannot? Are LLMs destined to merge with other modalities? Furthermore, could studying the distinctions between purely language-based models and multimodal models offer new insights into human cognition and representation in fields like cognitive science and psychology?

References

1. LeCun, Yann et al. (1989). « Handwritten digit recognition with a back-propagation network ». In: *Advances in neural information processing systems* 2.
2. Jumper, John et al. (2021). « Highly accurate protein structure prediction with AlphaFold ». In: *Nature* 596.7873, pp. 583–589.
3. Dosovitskiy, Alexey et al. (2020). « An image is worth 16x16 words: Transformers for image recognition at scale ». In: *arXiv preprint arXiv:2010.11929*.
4. Bendou, Yassir et al. (2022a). « Easy—ensemble augmented-shot-y-shaped learning: State-of-the-art few-shot classification with simple components ». In: *Journal of Imaging* 8.7, p. 179.
5. Snell, Jake, Kevin Swersky, and Richard Zemel (2017a). « Prototypical networks for few-shot learning ». In: *Advances in neural information processing systems* 30.
6. Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017a). « Model-agnostic meta-learning for fast adaptation of deep networks ». In: *International conference on machine learning*. PMLR, pp. 1126–1135.
7. Lafargue, Raphael et al. (2024a). « Oops, I Sampled it Again: Reinterpreting Confidence Intervals in Few-Shot Learning ». In: *Transactions on Machine Learning Research*. ISSN: 2835-8856. URL: https://openreview.net/forum?id=JxxkKt9yrx.
8. Triantafillou, Eleni et al. (2019). « Meta-dataset: A dataset of datasets for learning to learn from few examples ». In: *arXiv preprint arXiv:1903.03096*.
9. Luo, Xu et al. (2023). « A closer look at few-shot classification again ». In: *International Conference on Machine Learning*. PMLR, pp. 23103–23123.
10. Lafargue, Raphael et al. (2024b). *Few and Fewer: Learning Better from Few Examples Using Fewer Base Classes*. arXiv: 2401.15834 [cs.CV]. URL: https://arxiv.org/abs/2401.15834.
11. Du, Yilun et al. (2024). « Learning universal policies via text-guided video generation ». In: *Advances in Neural Information Processing Systems* 36.
12. Dakhel, Arghavan Moradi et al. (2023). « Github copilot ai pair programmer: Asset or liability? » In: *Journal of Systems and Software* 203, p. 111734.

13. Cetinic, Eva and James She (2022). « Understanding and creating art with AI: Review and outlook ». In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 18.2, pp. 1–22.
14. Peres, Ricardo Silva et al. (2020). « Industrial artificial intelligence in industry 4.0-systematic review, challenges and outlook ». In: *IEEE access* 8, pp. 220121–220139.
15. Carlo, Antonio (2021). « Artificial intelligence in the defence sector ». In: *Modelling and Simulation for Autonomous Systems: 7th International Conference, MESAS 2020, Prague, Czech Republic, October 21, 2020, Revised Selected Papers 7*. Springer, pp. 269–278.
16. McGuinness, Patrick (Year Published). *How AI changes everything*. Accessed on the 12th of June 2024. URL: https://patmcguinness.substack.com/.
17. « Amount of Data Created Daily (2024) » (n.d.). In: *https://explodingtopics.com/blog/data-generated-per-day* ().
18. Hu, Qi et al. (2011). « Toward improved aeromechanics simulations using recent advancements in scientific computing ». In: *Proceedings 67th Annual Forum of the American Helicopter Society*, pp. 3–5.
19. Wikipedia (2024). *Graphics processing unit — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=Graphics%20processing%20unit&oldid=1226357568. [Online; accessed 30-May-2024].
20. Sevilla, Jaime et al. (2022). « Compute trends across three eras of machine learning ». In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.
21. Prieto, Alberto et al. (n.d.). « Evolution of Computing Energy Efficiency: Koomey's Law Revisited ». In: *Available at SSRN 4743277* ().
22. Bahri, Yasaman et al. (2021). « Explaining neural scaling laws ». In: *arXiv preprint arXiv:2102.06701*.
23. Hestness, Joel et al. (2017). « Deep learning scaling is predictable, empirically ». In: *arXiv preprint arXiv:1712.00409*.
24. Hao, Karen (2019). « Training a single AI model can emit as much carbon as five cars in their lifetimes ». In: *MIT technology Review* 75, p. 103.
25. Wu, Carole-Jean et al. (2022). « Sustainable ai: Environmental implications, challenges and opportunities ». In: *Proceedings of Machine Learning and Systems* 4, pp. 795–813.
26. Heikkilaarchive, Melissa (2023). « AI's carbon footprint is bigger than you think ». In: *MIT Technology review*.

27. Luccioni, Alexandra Sasha, Yacine Jernite, and Emma Strubell (2024). « Power hungry processing: Watts driving the cost of ai deployment? » In: *ACM FAccT '24, June 3–6*.
28. Jiang, Albert Q et al. (2024). « Mixtral of experts ». In: *arXiv preprint arXiv:2401.04088*.
29. Bruckert, Sebastian, Bettina Finzel, and Ute Schmid (2020). « The next generation of medical decision support: A roadmap toward transparent expert companions ». In: *Frontiers in artificial intelligence* 3, p. 507973.
30. Schwalbe, Gesina and Bettina Finzel (2023). « A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts ». In: *Data Mining and Knowledge Discovery*, pp. 1–59.
31. Páez, Andrés (2019). « The pragmatic turn in explainable artificial intelligence (XAI) ». In: *Minds and Machines* 29.3, pp. 441–459.
32. Colin, Julien et al. (2022). « What i cannot predict, i do not understand: A human-centered evaluation framework for explainability methods ». In: *Advances in neural information processing systems* 35, pp. 2832–2845.
33. Zama (2022). *Concrete ML: a Privacy-Preserving Machine Learning Library using Fully Homomorphic Encryption for Data Scientists*. https://github.com/zama-ai/concrete-ml.
34. Lee, Joon-Woo et al. (2022). « Privacy-preserving machine learning with fully homomorphic encryption for deep neural network ». In: *iEEE Access* 10, pp. 30039–30054.
35. Zhang, Chen et al. (2021a). « A survey on federated learning ». In: *Knowledge-Based Systems* 216, p. 106775.
36. Grativol, Lucas et al. (2023). « Federated learning compression designed for lightweight communications ». In: *2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 1–4. DOI: 10.1109/ICECS58634.2023.10382717.
37. Hu, Hongsheng et al. (2022a). « Membership inference attacks on machine learning: A survey ». In: *ACM Computing Surveys (CSUR)* 54.11s, pp. 1–37.
38. Haim, Niv et al. (2022). « Reconstructing training data from trained neural networks ». In: *Advances in Neural Information Processing Systems* 35, pp. 22911–22924.
39. Dwork, Cynthia (2006). « Differential privacy ». In: *International colloquium on automata, languages, and programming*. Springer, pp. 1–12.
40. Turing, Alan Mathison (1950). « Computing Machinery and Intelligence ». In: *Mind* 49, pp. 433–460.

41. Csurka, Gabriela (2017). « Domain adaptation for visual applications: A comprehensive survey ». In: *arXiv preprint arXiv:1702.05374*.
42. Yang, Yao-Yuan et al. (2020a). « A closer look at accuracy vs. robustness ». In: *Advances in neural information processing systems* 33, pp. 8588–8601.
43. Tenenbaum, Joshua B et al. (2011). « How to grow a mind: Statistics, structure, and abstraction ». In: *science* 331.6022, pp. 1279–1285.
44. Robinson, Todd (2020). « Few-shot learning for defence and security ». In: *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*. Vol. 11413. SPIE, pp. 93–106.
45. Poulain, Raphael, Mehak Gupta, and Rahmatollah Beheshti (2022). « Few-shot learning with semi-supervised transformers for electronic health records ». In: *Machine Learning for Healthcare Conference*. PMLR, pp. 853–873.
46. O'shea, Keiron and Ryan Nash (2015). « An introduction to convolutional neural networks ». In: *arXiv preprint arXiv:1511.08458*.
47. *Conv2d 2014; PyTorch 2.3 documentation — pytorch.org* (n.d.). https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html. [Accessed 19-07-2024].
48. Vaswani, Ashish et al. (2017). « Attention is all you need ». In: *Advances in neural information processing systems* 30.
49. Cybenko, George (1989). « Approximation by superpositions of a sigmoidal function ». In: *Mathematics of control, signals and systems* 2.4, pp. 303–314.
50. Knut, Hinkelmann (2018). « Neural Networks p. 7 ». In: *University of Applied Sciences Northwestern Switzerland*.
51. Fukushima, Kunihiko (1969). « Visual feature extraction by a multilayered network of analog threshold elements ». In: *IEEE Transactions on Systems Science and Cybernetics* 5.4, pp. 322–333.
52. Hendrycks, Dan and Kevin Gimpel (2016). « Gaussian error linear units (gelus) ». In: *arXiv preprint arXiv:1606.08415*.
53. Maas, Andrew L, Awni Y Hannun, Andrew Y Ng, et al. (2013). « Rectifier nonlinearities improve neural network acoustic models ». In: *Proc. icml*. Vol. 30. 1. Atlanta, GA, p. 3.
54. Sutskever, Ilya et al. (2013). « On the importance of initialization and momentum in deep learning ». In: *International conference on machine learning*. PMLR, pp. 1139–1147.

55. Picard, David (2021). « Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision ». In: *arXiv preprint arXiv:2109.08203*.
56. He, Kaiming et al. (2016a). « Deep residual learning for image recognition ». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.
57. Telgarsky, Matus (2016). « Benefits of depth in neural networks ». In: *Conference on learning theory*. PMLR, pp. 1517–1539.
58. Radford, Alec et al. (2021). « Learning transferable visual models from natural language supervision ». In: *International conference on machine learning*. PMLR, pp. 8748–8763.
59. Chen, Ting et al. (2020). « A simple framework for contrastive learning of visual representations ». In: *International conference on machine learning*. PMLR, pp. 1597–1607.
60. Jaiswal, Ashish et al. (2020). « A survey on contrastive self-supervised learning ». In: *Technologies* 9.1, p. 2.
61. Bendou, Yassir et al. (2023). *Disambiguation of One-Shot Visual Classification Tasks: A Simplex-Based Approach*. arXiv: 2301.06372 [cs.CV]. URL: https://arxiv.org/abs/2301.06372.
62. Hu, Edward J et al. (2021). « Lora: Low-rank adaptation of large language models ». In: *arXiv preprint arXiv:2106.09685*.
63. Liu, Shih-Yang et al. (2024). « Dora: Weight-decomposed low-rank adaptation ». In: *arXiv preprint arXiv:2402.09353*.
64. Wang, Yan et al. (2019). « SimpleShot: Revisiting Nearest-Neighbor Classification for Few-Shot Learning ». In: *arXiv preprint arXiv:1911.04623*.
65. Garrido, Quentin et al. (2023). « Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank ». In: *International conference on machine learning*. PMLR, pp. 10929–10974.
66. Bendou, Yassir et al. (2022b). « Easy—Ensemble Augmented-Shot-Y-Shaped Learning: State-of-the-Art Few-Shot Classification with Simple Components ». In: *Journal of Imaging* 8.7. ISSN: 2313-433X. DOI: 10.3390/jimaging8070179. URL: https://www.mdpi.com/2313-433X/8/7/179.

67. Mangla, Puneet et al. (2020). « Charting the right manifold: Manifold mixup for few-shot learning ». In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2218–2227.
68. Chen, Da et al. (2021a). « Self-supervised learning for few-shot image classification ». In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1745–1749.
69. Yan, Shipeng, Songyang Zhang, Xuming He, et al. (2019). « A Dual Attention Network with Semantic Embedding for Few-Shot Learning. » In: *AAAI*, pp. 9079–9086.
70. Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017b). « Model-agnostic meta-learning for fast adaptation of deep networks ». In: *International Conference on Machine Learning*, pp. 1126–1135.
71. Munkhdalai, Tsendsuren et al. (2018). « Rapid adaptation with conditionally shifted neurons ». In: *International Conference on Machine Learning*, pp. 3664–3673.
72. Lee, Kwonjoon et al. (2019). « Meta-learning with differentiable convex optimization ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10657–10665.
73. Scott, Tyler, Karl Ridgeway, and Michael C Mozer (2018a). « Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning ». In: *Advances in Neural Information Processing Systems* 31.
74. Munkhdalai, Tsendsuren and Hong Yu (2017). « Meta networks ». In: *International Conference on Machine Learning*, pp. 2554–2563.
75. Zhang, Chi et al. (2021b). « Meta navigator: Search for a good adaptation policy for few-shot learning ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9435–9444.
76. Scott, Tyler R, Karl Ridgeway, and Michael C Mozer (2018b). « Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning ». In: *Advances in Neural Information Processing Systems* 2018-Decem, pp. 76–85.
77. Liu, Yanbin et al. (2018). « Learning to propagate labels: Transductive propagation network for few-shot learning ». In: *arXiv preprint arXiv:1805.10002*.
78. Bontonou, Myriam et al. (2021). « Few-Shot Decoding of Brain Activation Maps ». In: *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, pp. 1326–1330.

79. Henderson, Robert DE et al. (2021). « Automatic Detection and Classification of Multiple Catheters in Neonatal Radiographs with Deep Learning ». In: *Journal of Digital Imaging* 34.4, pp. 888–897.
80. Konstantin, Egorov et al. (2021). « Noise-resilient Automatic Interpretation of Holter ECG Recordings ». In: *BIOSIGNALS 2021-14th International Conference on Bio-Inspired Systems and Signal Processing; Part of the 14th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2021*, pp. 208–214.
81. Ma, Jiawei et al. (2021). « Partner-Assisted Learning for Few-Shot Image Classification ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10573–10582.
82. Luo, Xu et al. (2021a). « Rectifying the Shortcut Learning of Background for Few-Shot Learning ». In: *Advances in Neural Information Processing Systems* 34.
83. Snell, Jake, Kevin Swersky, and Richard Zemel (2017b). « Prototypical networks for few-shot learning ». In: *Advances in neural information processing systems* 30.
84. Zhang, Hongyi et al. (2018). « MixUp: Beyond empirical risk minimization ». In: *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. URL: https://github.com/facebookresearch/mixup-cifar10..
85. Verma, Vikas et al. (2019). « Manifold mixup: Better representations by interpolating hidden states ». In: *36th International Conference on Machine Learning, ICML 2019* 2019-June, pp. 11196–11205.
86. Li, Junjie, Zilei Wang, and Xiaoming Hu (2021). « Learning Intact Features by Erasing-Inpainting for Few-shot Classification ». In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.9, pp. 8401–8409.
87. Zhang, Chi et al. (2020). « DeepEMD: Few-Shot Image Classification With Differentiable Earth Mover's Distance and Structured Classifiers ». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12203–12213.
88. Choe, Junsuk et al. (2017). « Face generation for low-shot learning using generative adversarial networks ». In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1940–1948.
89. Li, Kai et al. (2020). « Adversarial feature hallucination networks for few-shot learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13470–13479.

90. Hariharan, Bharath and Ross Girshick (2017). « Low-shot visual recognition by shrinking and hallucinating features ». In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3018–3027.
91. Yang, Shuo, Lu Liu, and Min Xu (2021). « Free lunch for few-shot learning: Distribution calibration ». In: *arXiv preprint arXiv:2101.06395*.
92. Gidaris, Spyros et al. (2019). « Boosting few-shot visual learning with self-supervision ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8059–8068.
93. Ravi, Sachin and Hugo Larochelle (2017a). « Optimization as a model for few-shot learning ». In: *In International Confer- ence on Learning Representations (ICLR)*.
94. Vinyals, Oriol et al. (2016). « Matching networks for one shot learning ». In: *Advances in neural information processing systems* 29.
95. Liu, Jialin, Fei Chao, and Chih-Min Lin (2020). « Task augmentation by rotating for meta-learning ». In: *arXiv preprint arXiv:2003.00804*.
96. Luo, Xu et al. (2021b). « Boosting few-shot classification with view-learnable contrastive learning ». In: *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6.
97. Liu, Chen et al. (2021a). « Learning a Few-shot Embedding Model with Contrastive Learning ». In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.10, pp. 8635–8643.
98. Majumder, Orchid et al. (2021a). « Revisiting contrastive learning for few-shot classification ». In: *arXiv e-prints*, arXiv–2101.
99. Khosla, Prannay et al. (2020). « Supervised contrastive learning ». In: *Advances in Neural Information Processing Systems* 33, pp. 18661–18673.
100. Tian, Yonglong et al. (2020). « Rethinking few-shot image classification: a good embedding is all you need? » In: *European Conference on Computer Vision*, pp. 266–282.
101. Huang, Gao et al. (2017). « Snapshot ensembles: Train 1, get m for free ». In: *arXiv preprint arXiv:1704.00109*.
102. Chen, Wei Yu et al. (2019). « A closer look at few-shot classification ». In: *7th International Conference on Learning Representations, ICLR 2019* 2018, pp. 1–17.
103. Rodríguez, Pau et al. (2020). « Embedding propagation: Smoother manifold for few-shot classification ». In: *European Conference on Computer Vision*, pp. 121–138.

104. Hu, Yuqing, Vincent Gripon, and Stéphane Pateux (2021a). « Squeezing Backbone Feature Distributions to the Max for Efficient Few-Shot Learning ». In: *arXiv preprint arXiv:2110.09446*.

105. Loshchilov, Ilya and Frank Hutter (2016). « Sgdr: Stochastic gradient descent with warm restarts ». In: *arXiv preprint arXiv:1608.03983*.

106. He, Kaiming et al. (2016b). « Deep residual learning for image recognition ». In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-Decem, pp. 770–778.

107. Shorten, Connor and Taghi M Khoshgoftaar (2019). « A survey on image data augmentation for deep learning ». In: *Journal of big data* 6.1, pp. 1–48.

108. Oreshkin, Boris N, Pau Rodriguez, and Alexandre Lacoste (2018). « Tadam: Task dependent adaptive metric for improved few-shot learning ». In: *Advances in Neural Information Processing Systems* 2018-Decem, pp. 721–731.

109. Ye, Han-Jia et al. (2020). « Few-shot learning via embedding adaptation with set-to-set functions ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8808–8817.

110. Zhao, Jiabao et al. (2021). « Looking Wider for Better Adaptive Representation in Few-Shot Learning ». In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.12, pp. 10981–10989.

111. Fei, Nanyi et al. (2020). « MELR: Meta-learning via modeling episode-level relationships for few-shot learning ». In: *International Conference on Learning Representations*.

112. Rizve, Mamshad Nayeem et al. (2021). « Exploring Complementary Strengths of Invariant and Equivariant Representations for Few-Shot Learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10836–10846.

113. Boudiaf, Malik et al. (2020). « Information maximization for few-shot learning ». In: *Advances in Neural Information Processing Systems* 33, pp. 2445–2457.

114. Qi, Guodong et al. (2021). « Transductive Few-Shot Classification on the Oblique Manifold ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8412–8422.

115. Shen, Xi et al. (2021a). « Re-ranking for image retrieval and transductive few-shot classification ». In: *Advances in Neural Information Processing Systems* 34, pp. 25932–25943.

116. Lazarou, Michalis, Tania Stathaki, and Yannis Avrithis (2021). « Iterative label cleaning for transductive and semi-supervised few-shot learning ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8751–8760.
117. Yang, Ling et al. (June 2020b). « DPGN: Distribution Propagation Graph Network for Few-Shot Learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
118. Chen, Chaofan et al. (2021b). « ECKPN: Explicit Class Knowledge Propagation Network for Transductive Few-shot Learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6596–6605.
119. Le, Duong et al. (2021). « POODLE: Improving Few-shot Learning via Penalizing Out-of-Distribution Samples ». In: *Advances in Neural Information Processing Systems* 34, pp. 23942–23955.
120. Dhillon, Guneet S et al. (2019). « A baseline for few-shot image classification ». In: *arXiv preprint arXiv:1909.02729*.
121. Liu, Yaoyao, Bernt Schiele, and Qianru Sun (2020). « An ensemble of epoch-wise empirical bayes for few-shot learning ». In: *European Conference on Computer Vision*, pp. 404–421.
122. Hu, Yuqing, Vincent Gripon, and Stéphane Pateux (2021b). « Leveraging the feature distribution in transfer-based few-shot learning ». In: *International Conference on Artificial Neural Networks*, pp. 487–499.
123. Veilleux, Olivier et al. (2021). « Realistic evaluation of transductive few-shot learning ». In: *Advances in Neural Information Processing Systems* 34.
124. Wang, Yikai et al. (2020a). « Instance credibility inference for few-shot learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12836–12845.
125. Ziko, Imtiaz et al. (2020). « Laplacian regularized few-shot learning ». In: *International Conference on Machine Learning*, pp. 11660–11670.
126. Hu, Shell Xu et al. (2020). « Empirical bayes transductive meta-learning with synthetic gradients ». In: *arXiv preprint arXiv:2004.12696*.
127. Khattak, Muhammad Uzair et al. (2023). « Maple: Multi-modal prompt learning ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19113–19122.
128. Wang, Zhengbo et al. (2024). « A hard-to-beat baseline for training-free clip-based adaptation ». In: *arXiv preprint arXiv:2402.04087*.

129. Zhu, Xiangyang et al. (2023). « Not all features matter: Enhancing few-shot clip with adaptive prior refinement ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2605–2615.
130. Sung, Flood et al. (2018). « Learning to compare: Relation network for few-shot learning ». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1199–1208.
131. Arnold, Sébastien et al. (2021). « Uniform sampling over episode difficulty ». In: *Advances in Neural Information Processing Systems* 34, pp. 1481–1493.
132. Caron, Mathilde et al. (2021). *Emerging Properties in Self-Supervised Vision Transformers*. arXiv: 2104.14294 [cs.CV].
133. Zhou, Kaiyang et al. (2022a). « Learning to prompt for vision-language models ». In: *International Journal of Computer Vision* 130.9, pp. 2337–2348.
134. Lake, Brenden M, Ruslan Salakhutdinov, and Joshua B Tenenbaum (2015). « Human-level concept learning through probabilistic program induction ». In: *Science* 350.6266, pp. 1332–1338.
135. Maji, Subhransu et al. (2013). « Fine-grained visual classification of aircraft ». In: *arXiv preprint arXiv:1306.5151*.
136. Wah, Catherine et al. (2011). « The caltech-ucsd birds-200-2011 dataset ». In.
137. Cimpoi, Mircea et al. (2014). « Describing textures in the wild ». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613.
138. Schroeder, Brigit and Yin Cui (2018). « Fgvcx fungi classification challenge 2018 ». In: *Available online: github. com/visipedia/fgvcx_fungi_comp (accessed on 14 July 2021)*.
139. Nilsback, Maria-Elena and Andrew Zisserman (2008). « Automated flower classification over a large number of classes ». In: *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. IEEE, pp. 722–729.
140. Houben, Sebastian et al. (2013). « Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark ». In: *The 2013 international joint conference on neural networks (IJCNN)*. Ieee, pp. 1–8.
141. Jongejan, Jonas et al. (2016). « The quick, draw!-ai experiment ». In: *Mount View, CA, accessed Feb* 17.2018, p. 4.
142. Lin, Tsung-Yi et al. (2014). « Microsoft coco: Common objects in context ». In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, pp. 740–755.

143. Kumar, Ananya et al. (2022). « Fine-tuning can distort pretrained features and underperform out-of-distribution ». In: *arXiv preprint arXiv:2202.10054*.
144. Wang, Yaqing et al. (2020b). « Generalizing from a few examples: A survey on few-shot learning ». In: *ACM computing surveys (csur)* 53.3, pp. 1–34.
145. Antoniou, Antreas, Harrison Edwards, and Amos Storkey (2018). « How to train your MAML ». In: *arXiv preprint arXiv:1810.09502*.
146. Zhang, Renrui et al. (2021c). « Tip-adapter: Training-free clip-adapter for better vision-language modeling ». In: *arXiv preprint arXiv:2111.03930*.
147. Ren, Mengye et al. (2018). « Meta-learning for semi-supervised few-shot classification ». In: *arXiv preprint arXiv:1803.00676*.
148. Zhou, Kaiyang et al. (July 2022b). « Learning to Prompt for Vision-Language Models ». In: *International Journal of Computer Vision* 130.9, pp. 2337–2348. ISSN: 1573-1405. DOI: 10.1007/s11263-022-01653-1. URL: http://dx.doi.org/10.1007/s11263-022-01653-1.
149. Zhou, Yucan et al. (2020). « Expert training: Task hardness aware meta-learning for few-shot classification ». In: *arXiv preprint arXiv:2007.06240*.
150. Sun, Qianru et al. (2020). « Meta-transfer learning through hard tasks ». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.3, pp. 1443–1456.
151. Liu, Chenghao et al. (2020). « Adaptive task sampling for meta-learning ». In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*. Springer, pp. 752–769.
152. Student (1908). « The probable error of a mean ». In: *Biometrika* 6.1, pp. 1–25.
153. Hedberg, EC and Stephanie Ayers (2015). « The power of a paired t-test with a covariate ». In: *Social science research* 50, pp. 277–291.
154. Neyman, Jerzy (1937). « Outline of a theory of statistical estimation based on the classical theory of probability ». In: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 236.767, pp. 333–380.
155. DiCiccio, Thomas J and Bradley Efron (1996). « Bootstrap confidence intervals ». In: *Statistical science* 11.3, pp. 189–228.
156. Belia, Sarah et al. (2005). « Researchers misunderstand confidence intervals and standard error bars. » In: *Psychological methods* 10.4, p. 389.
157. Borji, Ali (2017). *Negative Results in Computer Vision: A Perspective*. arXiv: 1705.04402 [cs.CV].

158. Torralba, Antonio and Alexei A Efros (2011). « Unbiased look at dataset bias ». In: *CVPR 2011*. IEEE, pp. 1521–1528.
159. Zhou, Kaiyang et al. (2022c). « Conditional prompt learning for vision-language models ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16816–16825.
160. Oh, Jaehoon et al. (2022). « Understanding Cross-Domain Few-Shot Learning Based on Domain Similarity and Few-Shot Difficulty ». In: *Advances in Neural Information Processing Systems*.
161. Guo, Yunhui et al. (2020). « A broader study of cross-domain few-shot learning ». In: *European conference on computer vision*. Springer, pp. 124–141.
162. Kirillov, Alexander et al. (2023). « Segment anything ». In: *arXiv preprint arXiv:2304.02643*.
163. Wolpert, David H and William G Macready (1997). « No free lunch theorems for optimization ». In: *IEEE transactions on evolutionary computation* 1.1, pp. 67–82.
164. Bertinetto, Luca et al. (2018). « Meta-learning with differentiable closed-form solvers ». In: *arXiv preprint arXiv:1805.08136*.
165. Yoon, Sung Whan, Jun Seo, and Jaekyun Moon (2019). « TapNet: Neural network augmented with task-adaptive projection for few-shot learning ». In: *International Conference on Machine Learning*. PMLR, pp. 7115–7123.
166. Ravi, Sachin and Hugo Larochelle (2017b). « Optimization as a model for few-shot learning ». In: *International Conference on Learning Representations*.
167. Garcia Satorras, Victor and Joan Bruna Estrach (2018). « Few-shot learning with graph neural networks ». In: *International Conference on Learning Representations*.
168. Requeima, James et al. (2019). « Fast and flexible multi-task classification using conditional neural adaptive processes ». In: *Advances in Neural Information Processing Systems* 32.
169. Hou, Ruibing et al. (2019). « Cross attention network for few-shot classification ». In: *Advances in Neural Information Processing Systems* 32.
170. Doersch, Carl, Ankush Gupta, and Andrew Zisserman (2020). « CrossTransformers: Spatially-aware few-shot transfer ». In: *Advances in Neural Information Processing Systems* 33, pp. 21981–21993.
171. Bateni, Peyman et al. (2020). « Improved few-shot visual classification ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14493–14502.

172. Rebuffi, Sylvestre-Alvise, Hakan Bilen, and Andrea Vedaldi (2017). « Learning multiple visual domains with residual adapters ». In: *Advances in Neural Information Processing Systems*. Vol. 30.

173. Perez, Ethan et al. (2018). « FiLM: Visual reasoning with a general conditioning layer ». In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1.

174. Dvornik, Nikita, Cordelia Schmid, and Julien Mairal (2020). « Selecting relevant features from a multi-domain representation for few-shot classification ». In: *Computer Vision–ECCV 2020*. Springer, pp. 769–786.

175. Liu, Lu et al. (2021b). « A Universal Representation Transformer Layer for Few-Shot Image Classification ». In: *International Conference on Learning Representations*.

176. Li, Wei-Hong, Xialei Liu, and Hakan Bilen (2022). « Cross-domain few-shot learning with task-specific adapters ». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7161–7170.

177. Bertinetto, Luca et al. (2016). « Learning feed-forward one-shot learners ». In: *Advances in Neural Information Processing Systems* 29.

178. Oreshkin, Boris, Pau Rodríguez López, and Alexandre Lacoste (2018). « TADAM: Task dependent adaptive metric for improved few-shot learning ». In: *Advances in Neural Information Processing Systems* 31.

179. Garrido, Quentin et al. (2022). « RankMe: Assessing the downstream performance of pretrained self-supervised representations by their rank ». In: *arXiv preprint arXiv:2210.02885*.

180. Achille, Alessandro et al. (Oct. 2019). « Task2Vec: Task embedding for meta-learning ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6430–6439.

181. Shen, Zhiqiang et al. (2021b). « Partial is better than all: revisiting fine-tuning strategy for few-shot learning ». In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 11, pp. 9594–9602.

182. Su, Jong-Chyi, Subhransu Maji, and Bharath Hariharan (2020). « When does self-supervision improve few-shot learning? » In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*. Springer, pp. 645–666.

183. Majumder, Orchid et al. (2021b). « Supervised momentum contrastive learning for few-shot classification ». In: *arXiv preprint arXiv:2101.11058*.

184. Wang, Kafeng et al. (2020c). « Pay attention to features, transfer learn faster CNNs ». In: *International conference on learning representations*.

185. Islam, Ashraful et al. (2021). « Dynamic distillation network for cross-domain few-shot recognition with unlabeled data ». In: *Advances in Neural Information Processing Systems* 34, pp. 3584–3595.

186. Liu, Ziquan et al. (2021c). « Improved Fine-Tuning by Better Leveraging Pre-Training Data ». In: *Advances in Neural Information Processing Systems*.

187. Ge, Weifeng and Yizhou Yu (July 2017). « Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning ». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1086–1095.

188. Cui, Yin et al. (2018). « Large scale fine-grained categorization and domain-specific transfer learning ». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4109–4118.

189. Sahoo, Doyen et al. (2019). « Meta-learning with domain adaptation for few-shot learning under domain shift ». In.

190. Khandelwal, Pulkit and Paul Yushkevich (2020). « Domain generalizer: a few-shot meta learning framework for domain generalization in medical imaging ». In: *MICCAI Workshop: Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*. Springer, pp. 73–84.

191. Ward Jr, Joe H (1963). « Hierarchical grouping to optimize an objective function ». In: *Journal of the American statistical association* 58.301, pp. 236–244.

192. Bendou, Yassir et al. (2022c). « A Statistical Model for Predicting Generalization in Few-Shot Classification ». In: *arXiv preprint arXiv:2212.06461*.

193. Roy, Olivier and Martin Vetterli (2007). « The effective rank: A measure of effective dimensionality ». In: *2007 15th European signal processing conference*. IEEE, pp. 606–610.

194. Knight, Philip A (2008). « The Sinkhorn–Knopp algorithm: convergence and applications ». In: *SIAM Journal on Matrix Analysis and Applications* 30.1, pp. 261–275.

195. Rousseeuw, Peter J (1987). « Silhouettes: a graphical aid to the interpretation and validation of cluster analysis ». In: *Journal of computational and applied mathematics* 20, pp. 53–65.

196. Kingma, Diederik P and Jimmy Ba (2014). « Adam: A method for stochastic optimization ». In: *arXiv preprint arXiv:1412.6980*.

197. Polyak, Boris T (1964). « Some methods of speeding up the convergence of iteration methods ». In: *Ussr computational mathematics and mathematical physics* 4.5, pp. 1–17.

198. Bennequin, Etienne et al. (2021). « Bridging few-shot learning and adaptation: new challenges of support-query shift ». In: *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part I 21*. Springer, pp. 554–569.

199. Cordts, Marius et al. (2016). « The cityscapes dataset for semantic urban scene understanding ». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223.

200. Yang, Lihe et al. (2021). « Mining latent classes for few-shot segmentation ». In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8721–8730.

201. Hu, Shell Xu et al. (2022b). *Pushing the Limits of Simple Pipelines for Few-Shot Learning: External Data and Fine-Tuning Make a Difference*. arXiv: 2204.07305 [cs.CV]. URL: https://arxiv.org/abs/2204.07305.

202. Zha, Daochen et al. (2023). « Data-centric ai: Perspectives and challenges ». In: *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, pp. 945–948.

203. Feldman, Dan (2020). « Core-sets: Updated survey ». In: *Sampling techniques for supervised or unsupervised tasks*, pp. 23–44.

204. Massiceti, Daniela et al. (2021). « Orbit: A real-world few-shot dataset for teachable object recognition ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10818–10828.

205. Alex, Neel et al. (2021). « RAFT: A real-world few-shot text classification benchmark ». In: *arXiv preprint arXiv:2109.14076*.

206. Bennequin, Etienne (2023). « Challenges of Real Life Few-Shot Image Classification ». PhD thesis. Université Paris-Saclay.

207. Neau, Maëlic et al. (Oct. 2023). « Fine-Grained is Too Coarse: A Novel Data-Centric Approach for Efficient Scene Graph Generation ». In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 11–20.

208. Bensaid, Reda et al. (2024). *A Novel Benchmark for Few-Shot Semantic Segmentation in the Era of Foundation Models*. arXiv: 2401.11311 [cs.CV]. URL: https://arxiv.org/abs/2401.11311.

209. Gautam, Chandan et al. (2023). « Unsupervised Out-of-Distribution Detection Using Few in-Distribution Samples ». In: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1–5.

210. Elo, Arpad E (1967). « The proposed uscf rating system, its development, theory, and applications ». In: *Chess life* 22.8, pp. 242–247.

211. Jing, Longlong and Yingli Tian (2021). « Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey ». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11, pp. 4037–4058. DOI: 10.1109/TPAMI.2020.2992393.

212. Ben-Shaul, Ido et al. (2023). « Reverse engineering self-supervised learning ». In: *Advances in Neural Information Processing Systems* 36, pp. 58324–58345.

213. Khosla, Prannay et al. (2021). *Supervised Contrastive Learning*. arXiv: 2004.11362 [cs.LG]. URL: https://arxiv.org/abs/2004.11362.

214. Gandelsman, Yossi, Alexei A Efros, and Jacob Steinhardt (2023). « Interpreting CLIP's Image Representation via Text-Based Decomposition ». In: *arXiv preprint arXiv:2310.05916*.

215. Antol, Stanislaw et al. (2015). « Vqa: Visual question answering ». In: *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433.

216. Hossain, MD Zakir et al. (2019). « A comprehensive survey of deep learning for image captioning ». In: *ACM Computing Surveys (CsUR)* 51.6, pp. 1–36.

APPENDIX

5.4 Appendix of Chapter 2

5.4.1 Influence of the temperature in the transductive setting

Figure 5.1 shows how different values of the temperature β of the soft K-means changes the accuracy. $\beta = 5$ leads to the best accuracies on the two considered datasets. Consequently, we chose this value in our other experiments. In this experiment, we useed three ResNet12 with 30 augmented samples.

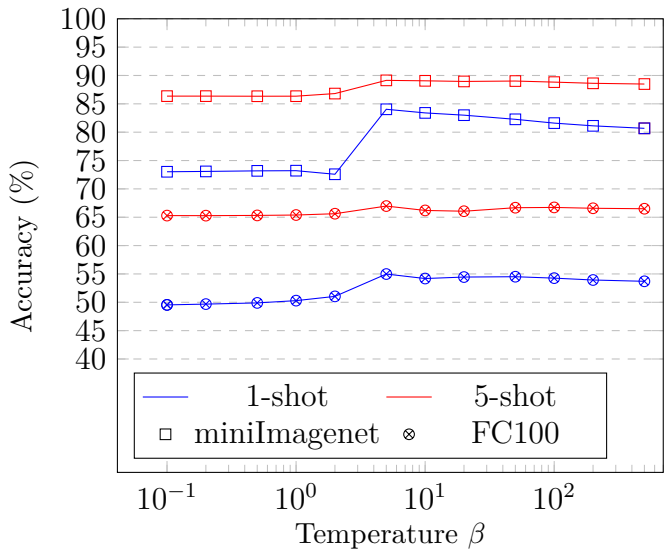


Figure 5.1 – Ablation study of Temperature of the soft K-means used in the transductive setting. We perform 10^5 runs for each value of β. (CC-BY)

5.4.2 Influence of the number of crops

In Figure 5.2, we show how the accuracy of our methodology is impacted by the number of crops ℓ used during Augmented Sampling (AS). When using $\ell = 1$, we report the performance of the method without using crops but using a simple global reshape

instead. The performance monotonically increases along with the number of crops used. A notable exception to this rule is a small drop of performance when switching from a global reshape to crops. This drop is easily explained simply as few crops are likely to miss the object of interest. However, the computational time to generate the crops also increases linearly. Consequently, we use $\ell = 30$ as a trade-off between accuracy and time complexity. We use a single ResNet12 in these experiments.

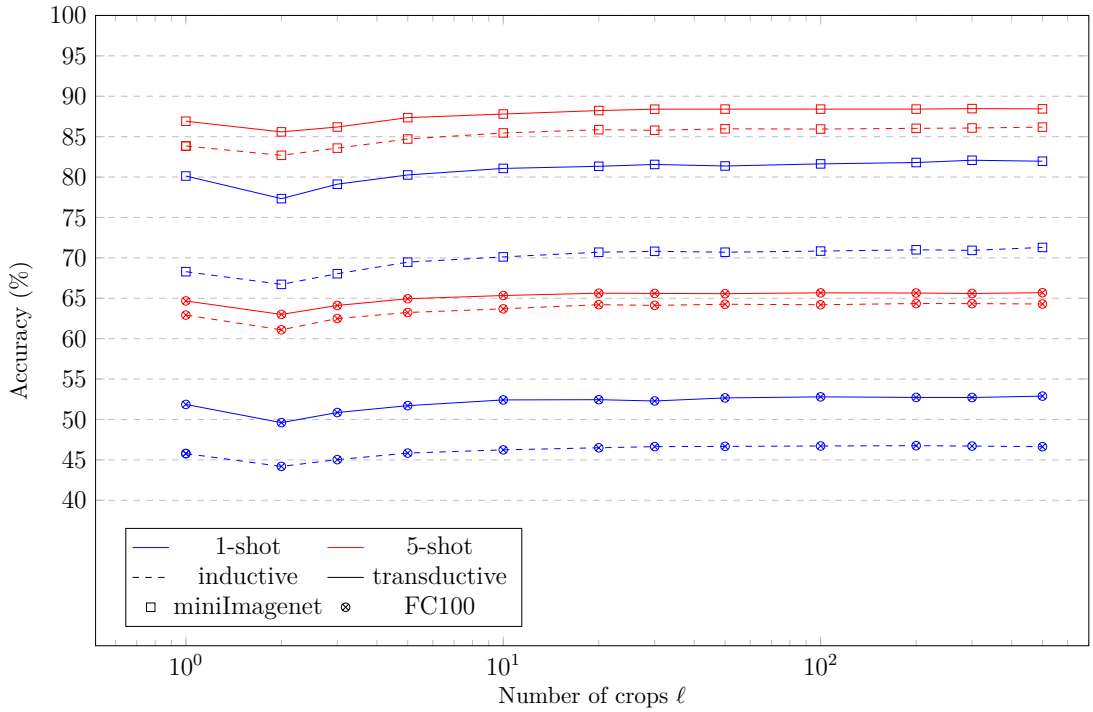


Figure 5.2 – Ablation study of Augmented Samples, we perform 10^5 runs for each value of ℓ. (CC-BY)

5.4.3 Influence of the number of backbones

Figure 5.3 shows how the performance is influenced by the number of backbones b at the Ensemble step (E). The performance increases steadily but saturates quickly. We use 30 augmented samples in this experiment.

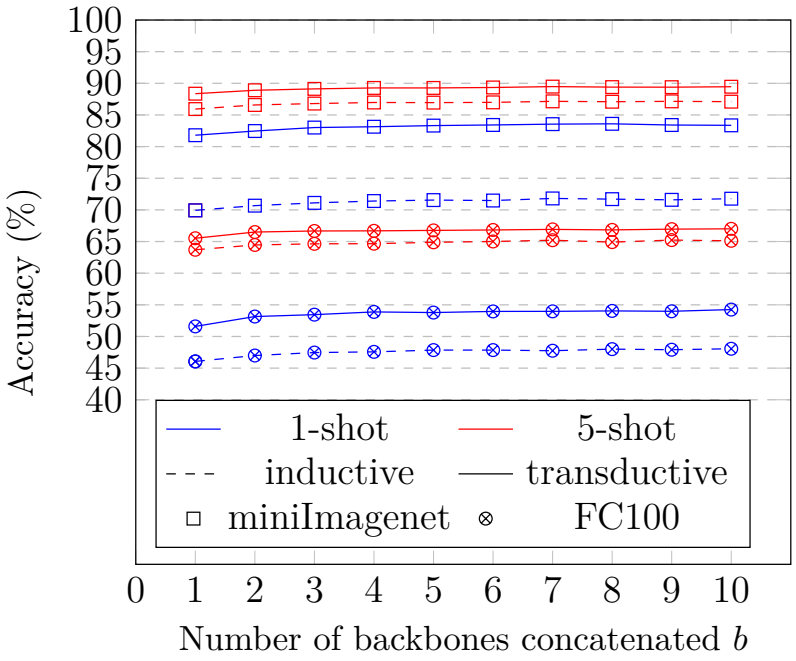


Figure 5.3 – Ablation study of the number of backbones, we perform 10^5 runs for each value of b. (CC-BY)

5.5 Appendix of Chapter 3: Additional results on the benchmark

Algorithm 4 Full dataset no-replacement evaluation algorithm

procedure EVALUATEUNTILDEPLETED($K, S, Q, \mathcal{C}, \{\mathcal{X}_c\}_{c\in\mathcal{C}}$) ▷ K ways, S shots, Q queries, set of classes C, set of data samples $\{\mathcal{X}_c\}_{c\in\mathcal{C}}$
  Initialize $\mathcal{T} = \{\}$ ▷ List to store all tasks
  while There are at least K classes in $\mathcal{C}$ with at least $S + Q$ examples each **do**
    $\mathcal{K} \leftarrow$ Randomly select K classes from $\mathcal{C}$ with at least $S + Q$ examples
    Initialize $\mathcal{S} = \{\}$ and $\mathcal{Q} = \{\}$
    for each c in $\mathcal{K}$ **do**
      $\mathcal{S}_c \leftarrow$ Randomly select S examples from $\mathcal{X}_c$
      $\mathcal{Q}_c \leftarrow$ Randomly select Q examples from $\mathcal{X}_c$ excluding $\mathcal{S}_c$
      Remove $\mathcal{S}_c$ and $\mathcal{Q}_c$ from $\mathcal{X}_c$
      Add $\mathcal{S}_c$ to $\mathcal{S}$
      Add $\mathcal{Q}_c$ to $\mathcal{Q}$
    end for
    Add $(\mathcal{S}, \mathcal{Q})$ to $\mathcal{T}$
  end while
  Initialize $\mathcal{A} = \{\}$ ▷ List to store all accuracies
  for $t \in \mathcal{T}$ **do**
    Add A_t to $\mathcal{A}$ ▷ Measure the accuracy of task t
  end for
  $\bar{A} = \text{MEAN}(\mathcal{A})$
  $\delta_{95\%} = t(|\mathcal{A}| - 1, 95\%)\sqrt{\frac{Var(\mathcal{A})}{|\mathcal{A}|}}$ ▷ t is the critical value for the Student't distribution
  return $\bar{A} \pm \delta_{95\%}$
end procedure

These results highlight the performance differences when using DINO and CLIP models with finetuning as baselines. Once again, finetuning generally underperforms compared to other adaptation methods.

Dataset	Method Sampling	LR	NCM
Aircraft	1-shot	-0.690 ± 1.790	0.690 ± 1.519
	5-shot	-5.286 ± 2.802	-2.857 ± 2.731
	10-shot	-5.510 ± 3.391	-0.408 ± 3.290
CUB	1-shot	-1.486 ± 1.467	0.914 ± 1.388
	5-shot	-2.182 ± 1.740	-0.970 ± 1.738
	10-shot	-4.874 ± 2.737	-3.529 ± 2.822
DTD	1-shot	-1.220 ± 2.111	-0.488 ± 1.540
	5-shot	-1.750 ± 2.457	-0.250 ± 2.518
	10-shot	-2.540 ± 1.320	-1.587 ± 1.937
Fungi	1-shot	-0.549 ± 0.561	1.427 ± 0.514
	5-shot	-3.166 ± 0.728	-1.770 ± 0.627
	10-shot	-3.658 ± 0.920	-3.009 ± 0.789
MSCOCO	1-shot	0.840 ± 0.389	0.000 ± 0.380
	5-shot	-0.846 ± 0.199	-0.210 ± 0.176
	10-shot	-2.488 ± 0.224	-0.695 ± 0.193
Omniglot	1-shot	2.976 ± 0.675	2.749 ± 0.724
	5-shot	1.430 ± 0.484	-0.183 ± 0.556
	10-shot	0.065 ± 0.560	-0.458 ± 0.656
Quickdraw	1-shot	1.800 ± 0.646	3.590 ± 0.691
	5-shot	-0.546 ± 0.267	-0.680 ± 0.303
	10-shot	-1.841 ± 0.223	-1.531 ± 0.260
Traffic Signs	1-shot	0.880 ± 0.429	1.310 ± 0.486
	5-shot	-0.900 ± 0.354	0.911 ± 0.372
	10-shot	-2.189 ± 0.400	0.725 ± 0.412
VGG Flower	1-shot	-2.542 ± 1.755	1.695 ± 1.475
	5-shot	-0.545 ± 0.623	-0.909 ± 1.085
	10-shot	-0.519 ± 0.776	-0.519 ± 0.776

Table 5.1 – Paired test difference between DINO with FT and LR and NCC on DINO. FT, NCC and LR respectively stand for Fine-tuning, Nearest Class Centroid, Logistic Regression. (CC-BY)

Dataset	Method Sampling	LR	NCM
Aircraft	1-shot	0.828 ± 1.850	4.000 ± 2.033
	5-shot	0.000 ± 1.934	0.571 ± 2.301
	10-shot	-3.469 ± 2.436	-5.510 ± 2.191
CUB	1-shot	-0.229 ± 1.500	1.714 ± 1.313
	5-shot	-0.364 ± 0.962	-0.848 ± 1.266
	10-shot	-0.504 ± 0.934	-0.504 ± 1.297
DTD	1-shot	-2.195 ± 2.184	2.195 ± 2.584
	5-shot	-2.500 ± 3.002	-2.000 ± 2.806
	10-shot	-5.079 ± 2.859	-4.444 ± 1.937
Fungi	1-shot	-2.086 ± 0.747	0.439 ± 0.703
	5-shot	-3.217 ± 0.792	-2.826 ± 0.868
	10-shot	-4.394 ± 0.929	-4.069 ± 0.986
MSCOCO	1-shot	-0.840 ± 0.536	1.670 ± 0.528
	5-shot	-2.090 ± 0.241	-0.970 ± 0.232
	10-shot	-3.099 ± 0.239	-1.503 ± 0.233
Omniglot	1-shot	4.723 ± 0.725	4.723 ± 0.799
	5-shot	1.688 ± 0.606	0.532 ± 0.670
	10-shot	0.371 ± 0.702	-0.087 ± 0.831
Quickdraw	1-shot	0.760 ± 0.731	4.940 ± 0.759
	5-shot	0.226 ± 0.299	-0.284 ± 0.327
	10-shot	-0.376 ± 0.236	-0.754 ± 0.268
Traffic Signs	1-shot	-0.500 ± 0.590	0.590 ± 0.606
	5-shot	-0.481 ± 0.373	-0.745 ± 0.449
	10-shot	-1.642 ± 0.418	-0.458 ± 0.478
VGG Flower	1-shot	-1.356 ± 1.408	0.169 ± 1.305
	5-shot	-0.182 ± 0.665	0.000 ± 0.774
	10-shot	-0.519 ± 0.776	-0.519 ± 0.776

Table 5.2 – Paired test difference between CLIP with FT and LR and NCC on CLIP. FT, NCC and LR respectively stand for Fine-tuning, Nearest Class Centroid, Logistic Regression. (CC-BY)

5.6 Appendix of Chapter 4

5.6.1 Other tables and Figures

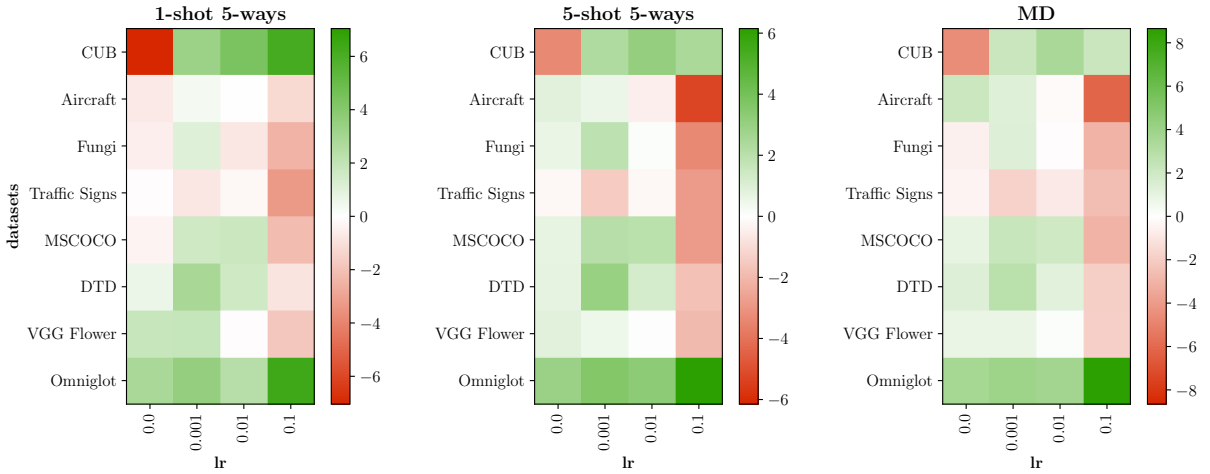


Figure 5.4 – Boost in Accuracy compared to the baseline for various learning rates **lr** using the DI selected feature extractor of each dataset. Learning rate is set to 0 when only batch normalization statistics are updated. In the chapter we only show the case of $lr = 0.001$. We observe a significant effect of the choice of the learning rate. (CC-BY)

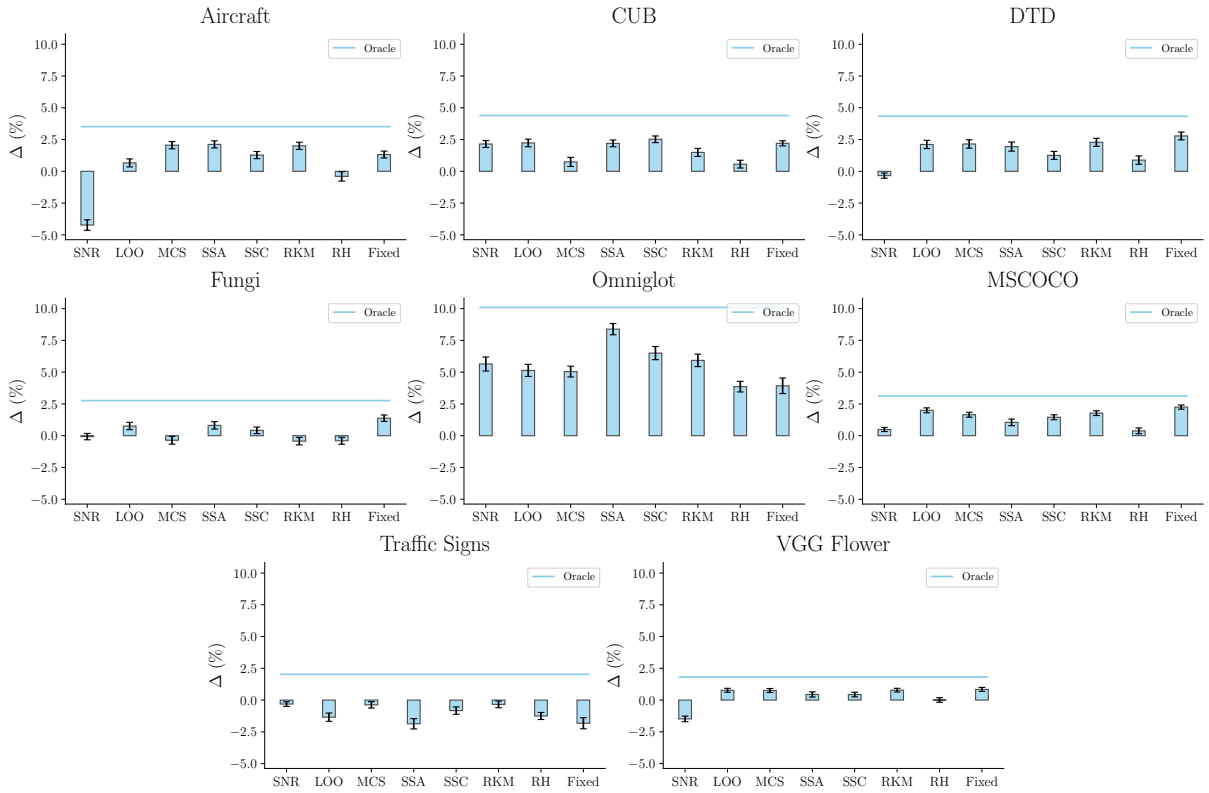


Figure 5.5 – Selection of learning rate in DI setting using heuristics in MD sampling. Fixed corresponds to the performance of lr = 0.001 that was presented in the first table of the chapter. Our methods outperforms the DI accuracy boost (Fixed) on Aircraft, Omniglot and Traffic Signs. We use the different learning rates presented in Table 5.4 (CC-BY)

Table 5.3 – Performance change using the fine-tuning on the support (S), with a Task-Informed (TI) subset selection, a Domain-Informed (DI) subset selection, and DI-UOT subset selection. All positive boosts with overlapping confidence intervals are bolded. (CC-BY)

Dataset	Method	1-shot 5-ways		5-shots 5-ways		MD	
		Baseline	Δ	Baseline	Δ	Baseline	Δ
Aircraft	S	39.95 ±0.70	-3.60 ±0.64	63.18 ±0.74	-1.48 ±0.61	65.86 ±0.90	**+5.33 ±0.69**
	TI		-0.06 ±0.33		**+0.26 ±0.31**		+1.33 ±0.25
	DI		**+0.34 ±0.32**		**+0.54 ±0.31**		+1.32 ±0.27
	DI-UOT		-0.25 ±0.33		-0.04 ±0.30		+0.86 ±0.27
	TI-UOT		-0.06 ±0.33		-0.01 ±0.30		+0.93 ±0.26
CUB	S	64.34 ±0.90	-19.28 ±0.88	87.78 ±0.59	-18.97 ±0.63	79.29 ±0.90	-14.51 ±0.60
	TI		+2.64 ±0.44		**+2.16 ±0.26**		+1.08 ±0.19
	DI		**+3.27 ±0.44**		**+2.29 ±0.26**		**+2.20 ±0.20**
	DI-UOT		+2.99 ±0.43		**+2.07 ±0.27**		**+1.97 ±0.20**
	TI-UOT		+2.64 ±0.44		+1.11 ±0.26		+0.96 ±0.19
DTD	S	45.21 ±0.77	+0.66 ±0.77	70.10 ±0.59	-3.12 ±0.59	76.03 ±0.69	-6.67 ±0.69
	TI		**+2.85 ±0.46**		**+2.77 ±0.33**		**+2.44 ±0.29**
	DI		**+2.90 ±0.48**		**+2.96 ±0.33**		**+2.78 ±0.31**
	DI-UOT		**+2.26 ±0.51**		**+2.62 ±0.34**		**+2.82 ±0.32**
	TI-UOT		**+2.85 ±0.46**		**+2.44 ±0.32**		**+2.82 ±0.32**
Fungi	S	53.01 ±0.92	-6.59 ±0.74	74.87 ±0.80	-8.33 ±0.62	51.57 ±1.16	-15.05 ±0.53
	TI		**+0.92 ±0.39**		**+1.67 ±0.30**		**+1.07 ±0.26**
	DI		**+1.07 ±0.41**		**+1.89 ±0.29**		**+1.38 ±0.25**
	DI-UOT		+0.74 ±0.40		**+1.46 ±0.29**		**+0.91 ±0.25**
	TI-UOT		**+0.92 ±0.39**		**+1.51 ±0.28**		+0.80 ±0.25
Omniglot	S	61.80 ±1.03	-3.16 ±1.11	81.53 ±0.76	**+3.53 ±0.85**	59.51 ±1.31	-4.59 ±1.07
	TI		**+2.65 ±0.38**		**+2.94 ±0.29**		**+3.74 ±0.23**
	DI		**+3.52 ±1.22**		**+3.57 ±0.81**		**+3.93 ±0.61**
	DI-UOT		-3.70 ±1.00		-5.02 ±0.68		-5.76 ±0.66
	TI-UOT		**+2.65 ±0.38**		+2.58 ±0.82		+2.46 ±0.62
MSCOCO	S	43.91 ±0.85	-5.44 ±0.66	63.04 ±0.79	-6.20 ±0.63	44.99 ±0.99	-17.00 ±0.72
	TI		**+1.27 ±0.35**		**+1.87 ±0.29**		+1.85 ±0.17
	DI		**+1.62 ±0.34**		**+2.09 ±0.30**		**+2.25 ±0.17**
	DI-UOT		**+1.27 ±0.35**		**+1.75 ±0.29**		**+2.09 ±0.18**
	TI-UOT		**+1.27 ±0.35**		+1.30 ±0.28		**+2.05 ±0.18**
Traffic Signs	S	**57.35 ±0.85**	-4.67 ±0.66	74.11 ±0.78	**+6.17 ±0.62**	**53.77 ±1.05**	**+0.77 ±1.00**
	TI		**-0.84 ±0.32**		-1.22 ±0.25		-2.02 ±0.17
	DI		**-0.79 ±0.95**		-1.48 ±0.77		-1.82 ±0.44
	DI-UOT		**-0.48 ±0.33**		-0.64 ±0.27		-1.26 ±0.18
	TI-UOT		**-0.84 ±0.32**		-0.99 ±0.77		-1.33 ±0.43
VGG Flower	S	75.86 ±0.84	+0.19 ±0.79	94.46 ±0.33	-1.45 ±0.37	92.77 ±0.58	-5.18 ±0.51
	TI		**+2.04 ±0.40**		**+0.64 ±0.18**		**+1.03 ±0.16**
	DI		**+1.88 ±0.41**		**+0.52 ±0.18**		**+0.84 ±0.16**
	DI-UOT		**+2.18 ±0.40**		**+0.67 ±0.18**		**+0.90 ±0.16**
	TI-UOT		**+2.04 ±0.40**		**+0.90 ±0.17**		**+0.95 ±0.16**
Average	S		-5.24 ±0.78		-3.73 ±0.61		-7.11 ±0.73
	TI		**+1.43 ±0.38**		**+1.39 ±0.28**		**+1.31 ±0.21**
	DI		**+1.73 ±0.57**		**+1.55 ±0.41**		**+1.61 ±0.30**
	DI-UOT		+0.63 ±0.47		+0.36 ±0.33		+0.32 ±0.28
	TI-UOT		**+1.43 ±0.36**		**+1.10 ±0.44**		**+1.21 ±0.32**

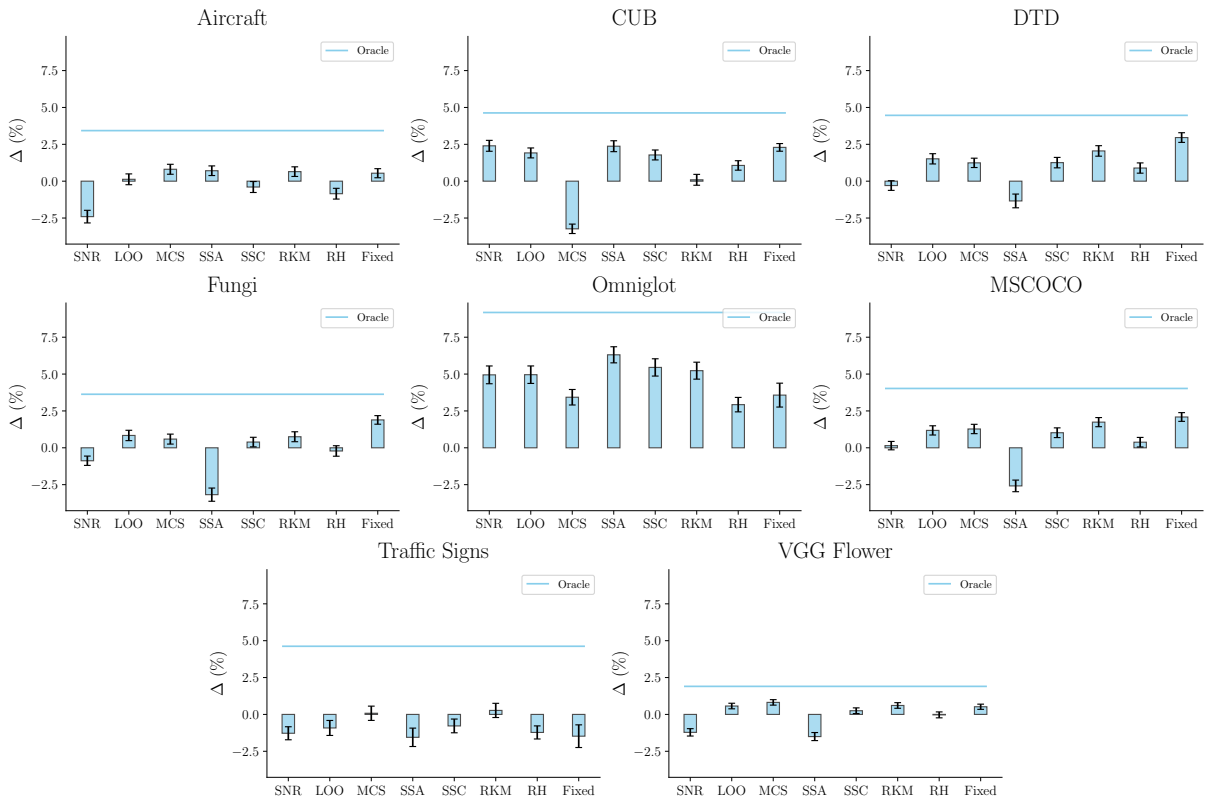


Figure 5.6 – Selection of learning rate in DI setting using heuristics in 5-ways 5-shots sampling. Fixed corresponds to the performance of lr = 0.001 that was presented in the first table of the chapter. Our methods outperforms the DI accuracy boost (Fixed) on Aircraft, Omniglot and Traffic Signs. We use the different learning rates presented in Table 5.4 (CC-BY)

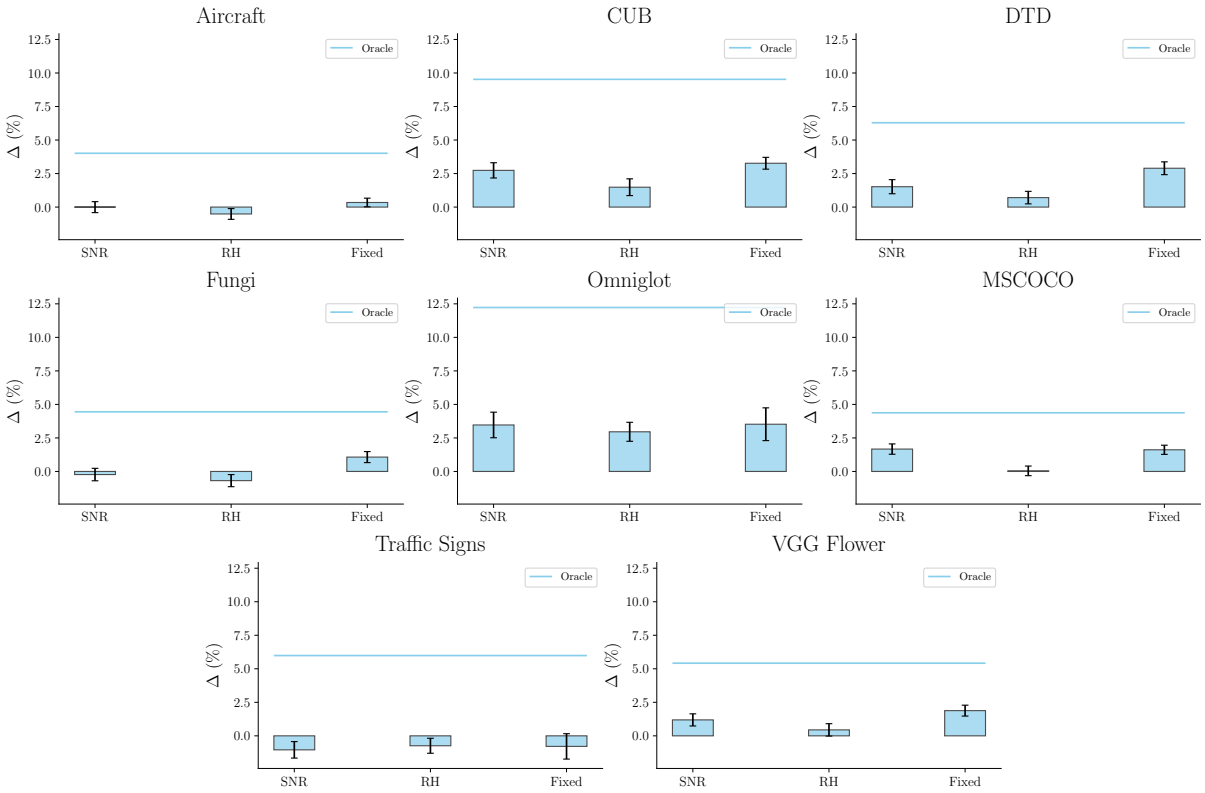


Figure 5.7 – Selection of learning rate in DI setting using heuristics in 1-ways 5-shots sampling. Fixed corresponds to the performance of lr = 0.001 that was presented in the first table of the chapter. In this case, the available data for the selection is not sufficient to outperform *Fixed*. We use the different learning rates presented in Table 5.4 (CC-BY)

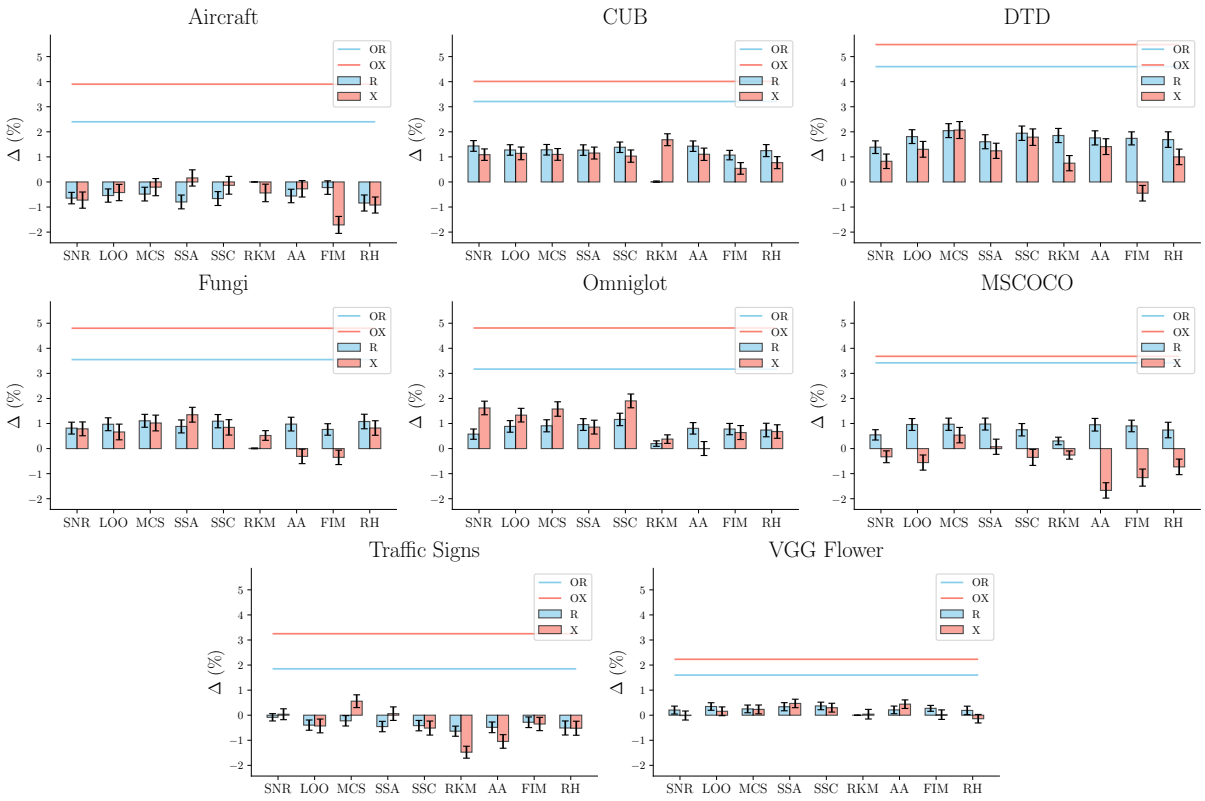


Figure 5.8 – Difference of accuracy with baseline after feature extractor selection using heuristics. Task are sampled following the 5-ways 5-shots sampling procedure. In R (resp. X) heuristics select a feature extractor amongst the R (resp. X) library of feature extractor. The oracle OR (resp. OX) selects the best feature extractor for each task in the R (resp. X) library. The Random Heuristic (RH) picks a feature extractor uniformly at random. (CC-BY)

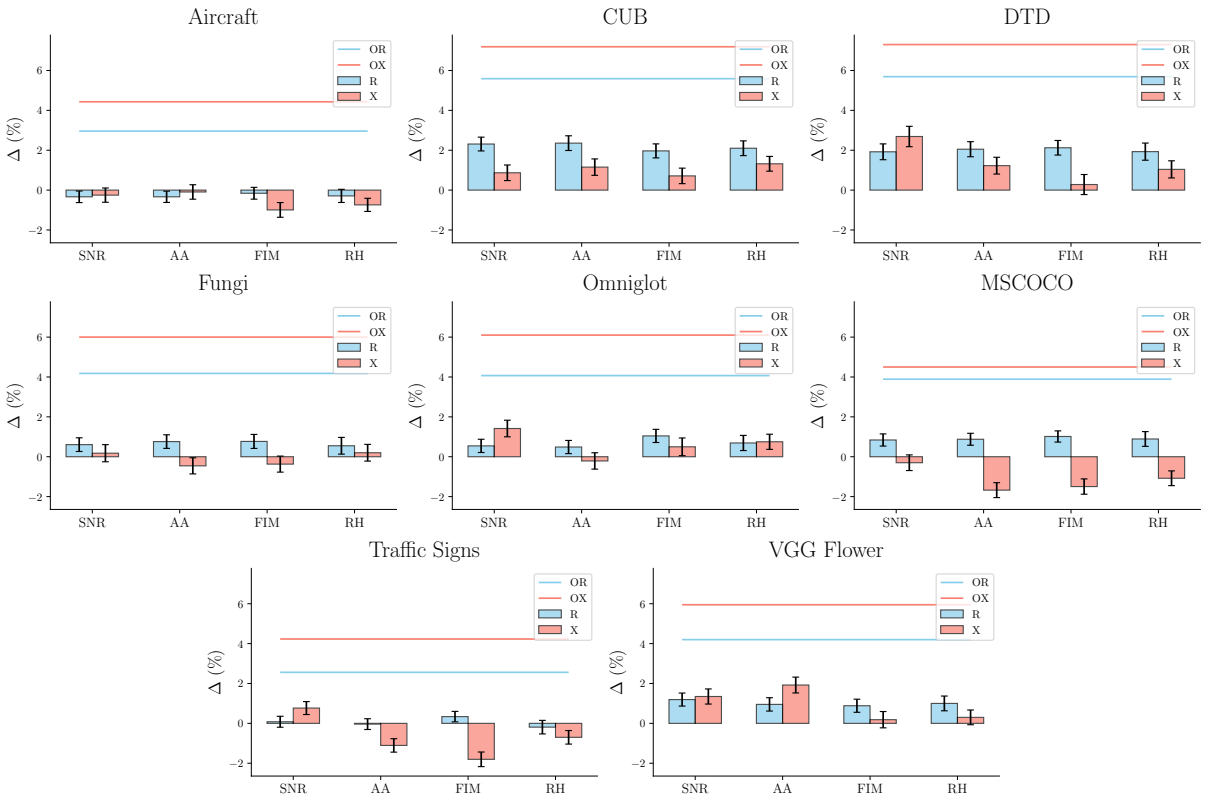


Figure 5.9 – Difference of accuracy with baseline after feature extractor selection using heuristics. Task are sampled following the 1-ways 5-shots sampling procedure. In R (resp. X) heuristics select a feature extractor amongst the R (resp. X) library of feature extractor. The oracle OR (resp. OX) selects the best feature extractor for each task in the R (resp. X) library. The Random Heuristic (RH) picks a feature extractor uniformly at random. (CC-BY)

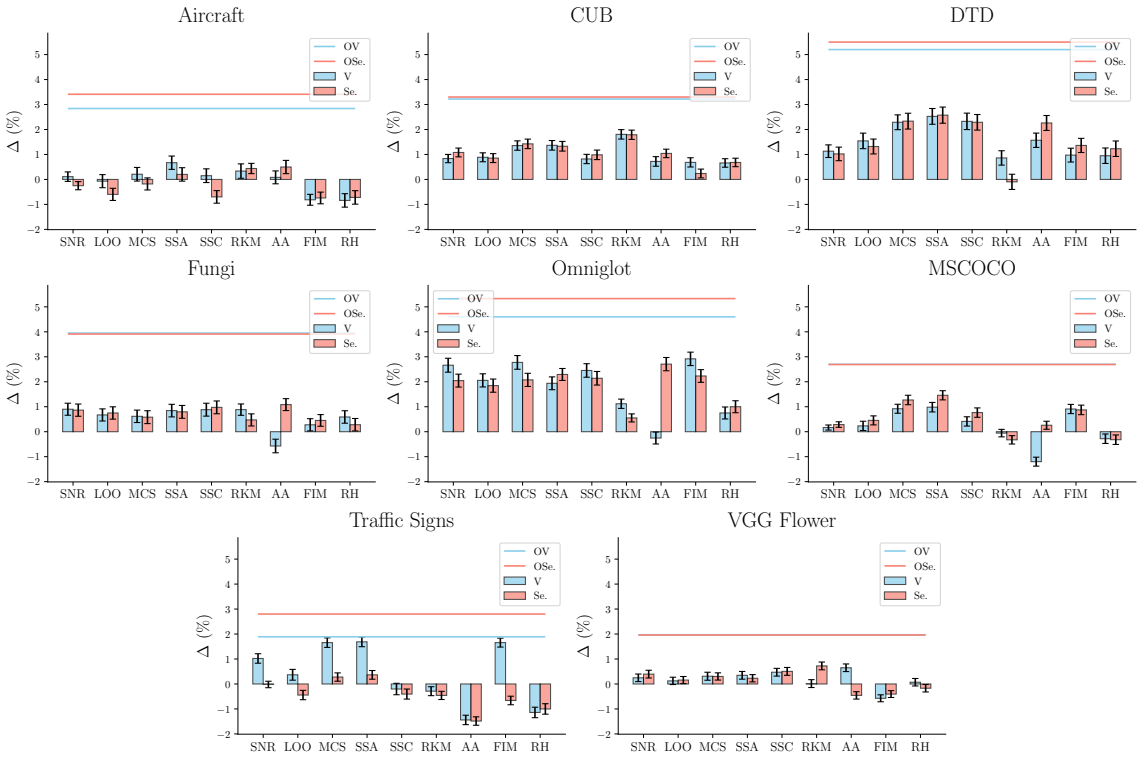


Figure 5.10 – Difference of accuracy with baseline after feature extractor selection using heuristics. Task are sampled following the MD protocol. In V (resp. Se.) heuristics select a feature extractor amongst the V (resp. Se.) library of feature extractor. The oracle OV (resp. OSe.) selects the best feature extractor for each task in the V (resp .Se.) library. The Random Heuristic (RH) picks a random feature extractor. (CC-BY)

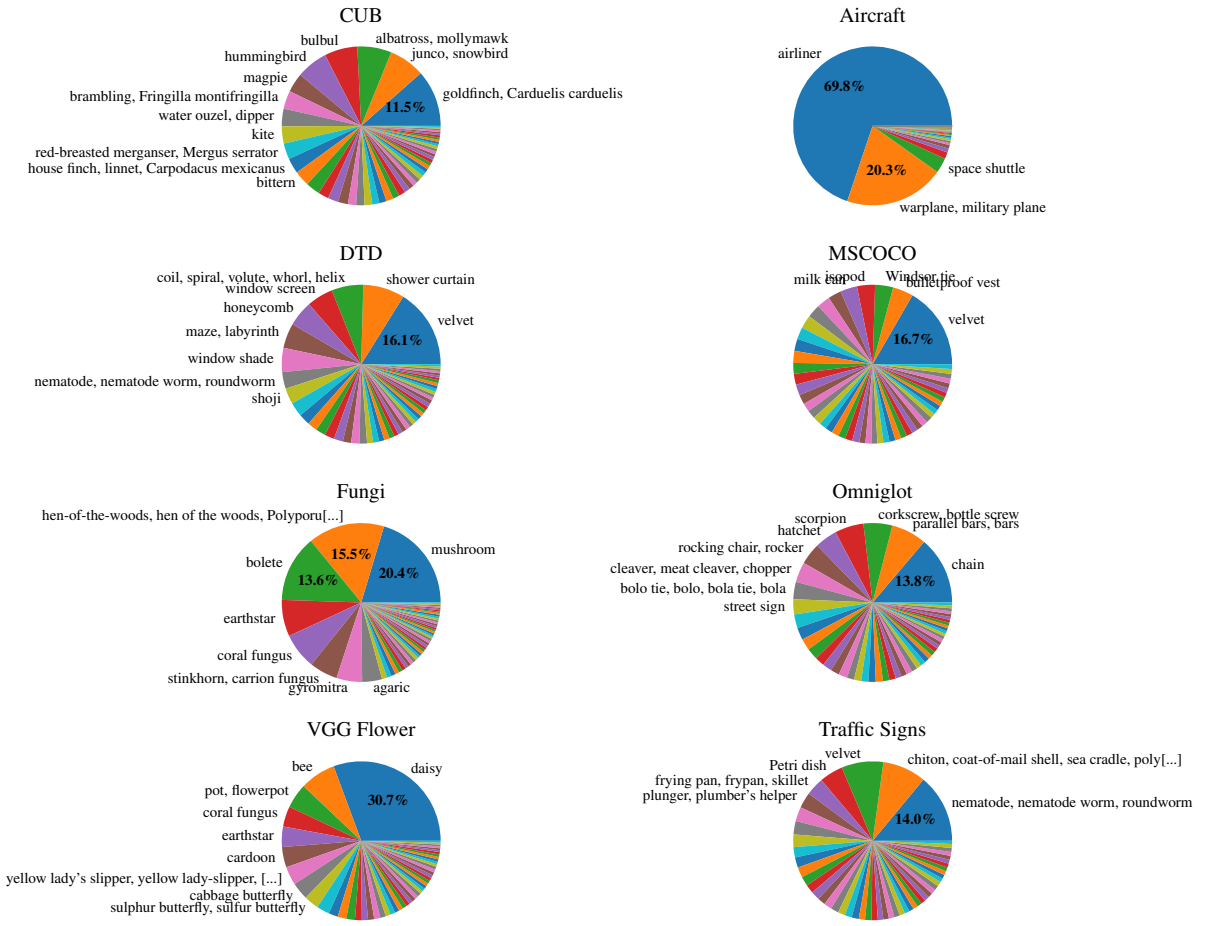


Figure 5.11 – Logit activations of ImageNet classes when target datasets are processed by the base model. While it may seem surprising that the ImageNet 'Street sign' class is not strongly activated within the Traffic Signs dataset, this is because its tightly cropped, low resolution images are highly dissimilar from the photographs of street signs in ImageNet. Notice how Aircraft is almost fully captured by two classes. (CC-BY)

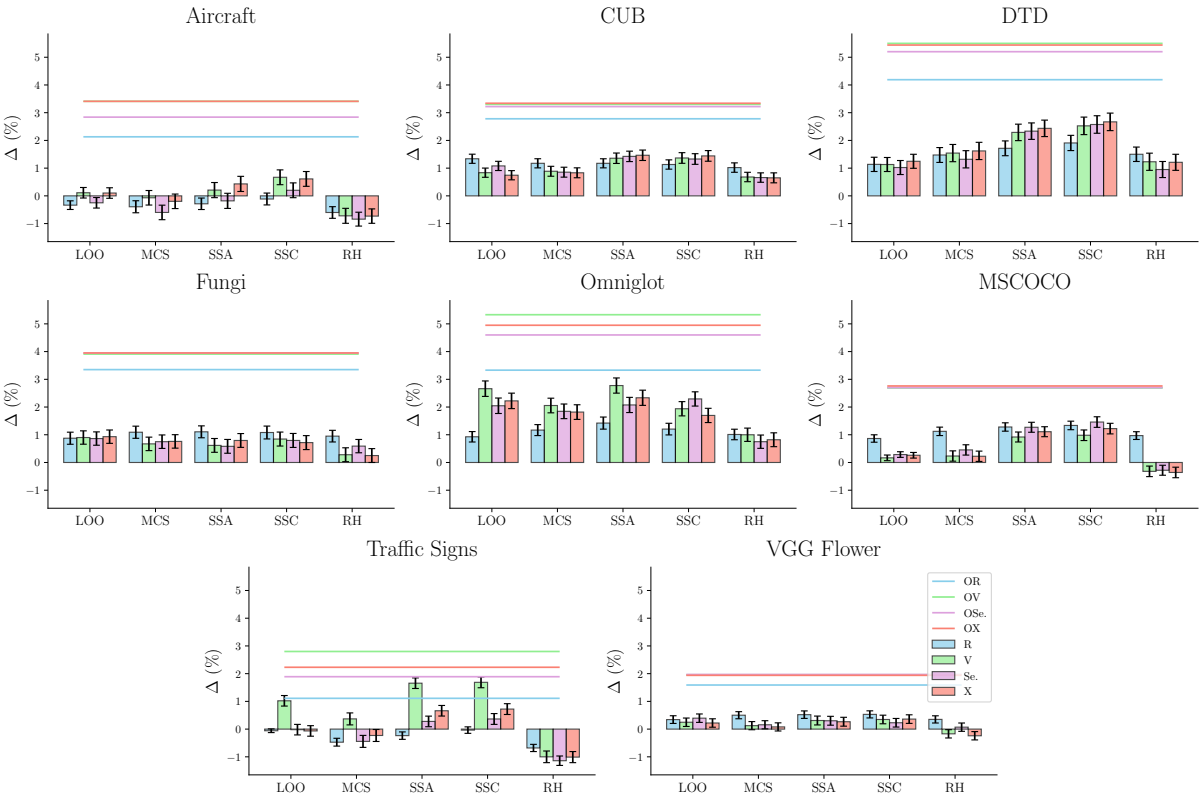


Figure 5.12 – Ablation of the effect of R, V, Se. and X on a reduced number of heuristics for readability. X is sometimes outperformed by V and Se. but overall X is the best. (CC-BY)

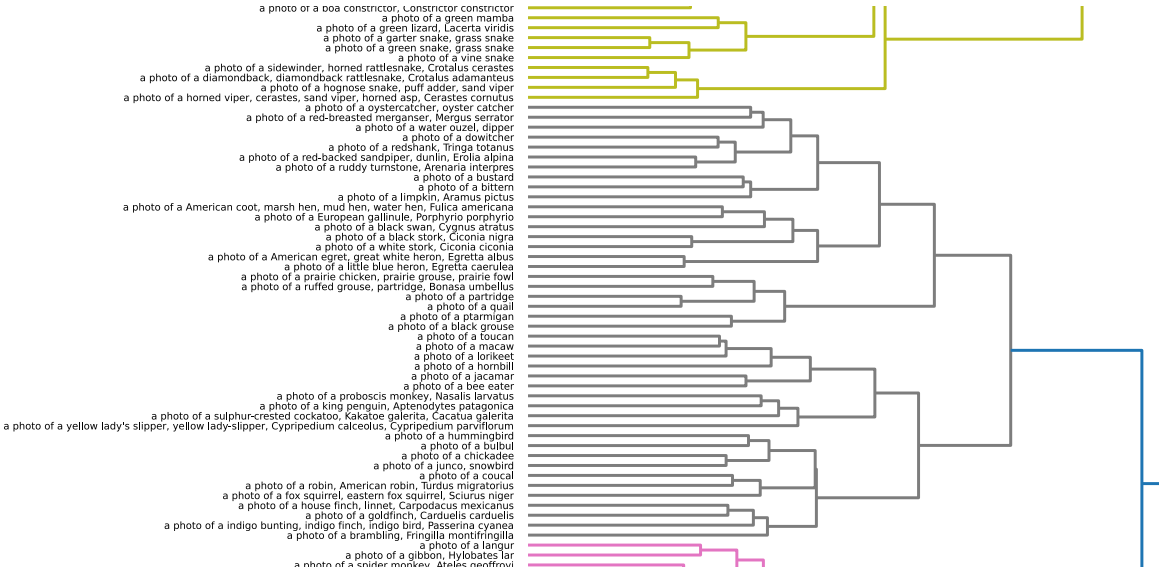


Figure 5.13 – Zoom over the birds (gray), reptiles (yellow) and monkeys (pink) of the Semantic (Se.) dendrogram of classes built using Ward's method. Notice that we used "a photo of a" in front of each classes to improve the CLIP embedding. Out of the 44 classes in the birds cluster 3 classes are not birds : The proboscis monkey, Yellow Lady's Slipper, Fox Squirrel. Their semantic relations to birds must explain their relation to this cluster. Some ambiguous words like "Crane" or "Kite" might not be captured by the semantic embeddings. (CC-BY)

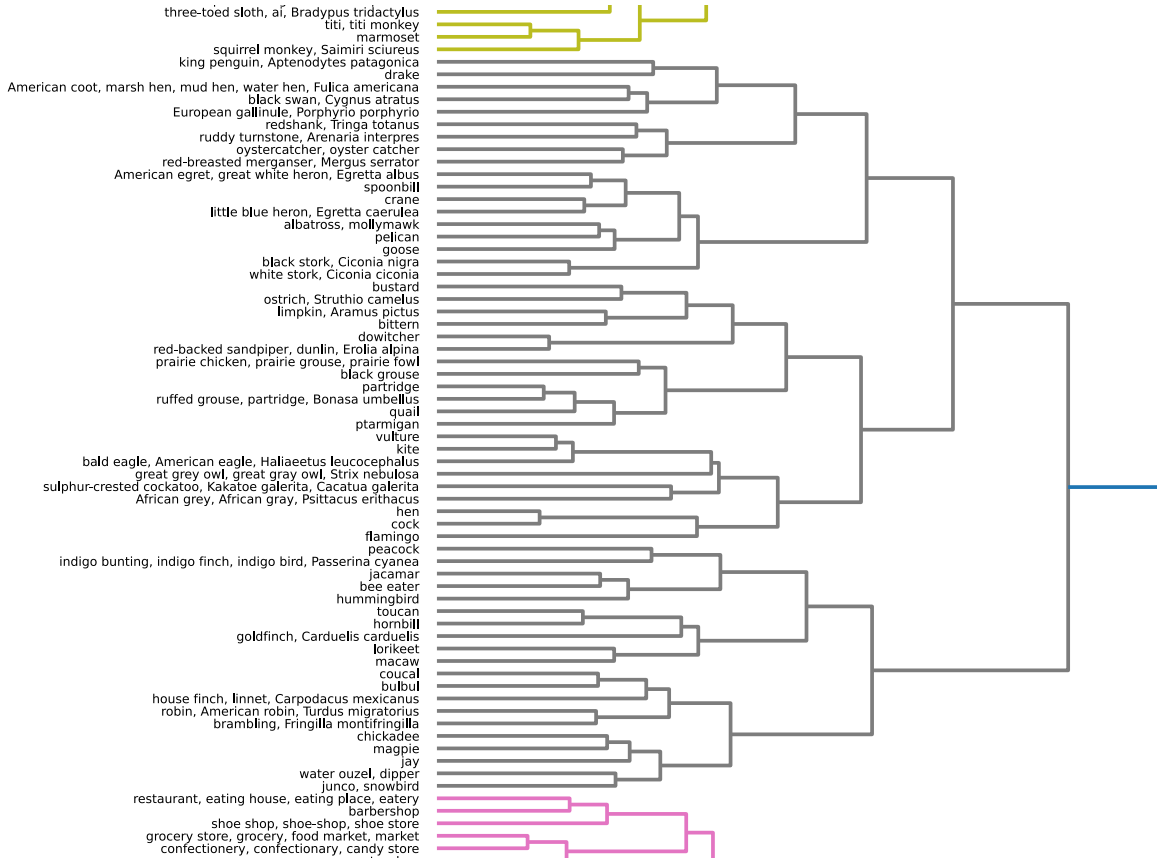


Figure 5.14 – Zoom over the birds (gray), reptiles (yellow) and buildings (pink) of the Visual (Se.) dendrogram of classes built using Ward's method. Notice that "Kite" was classified as part of the birds for trivial reasons (kites in the sky can be mistaken for a bird). (CC-BY)

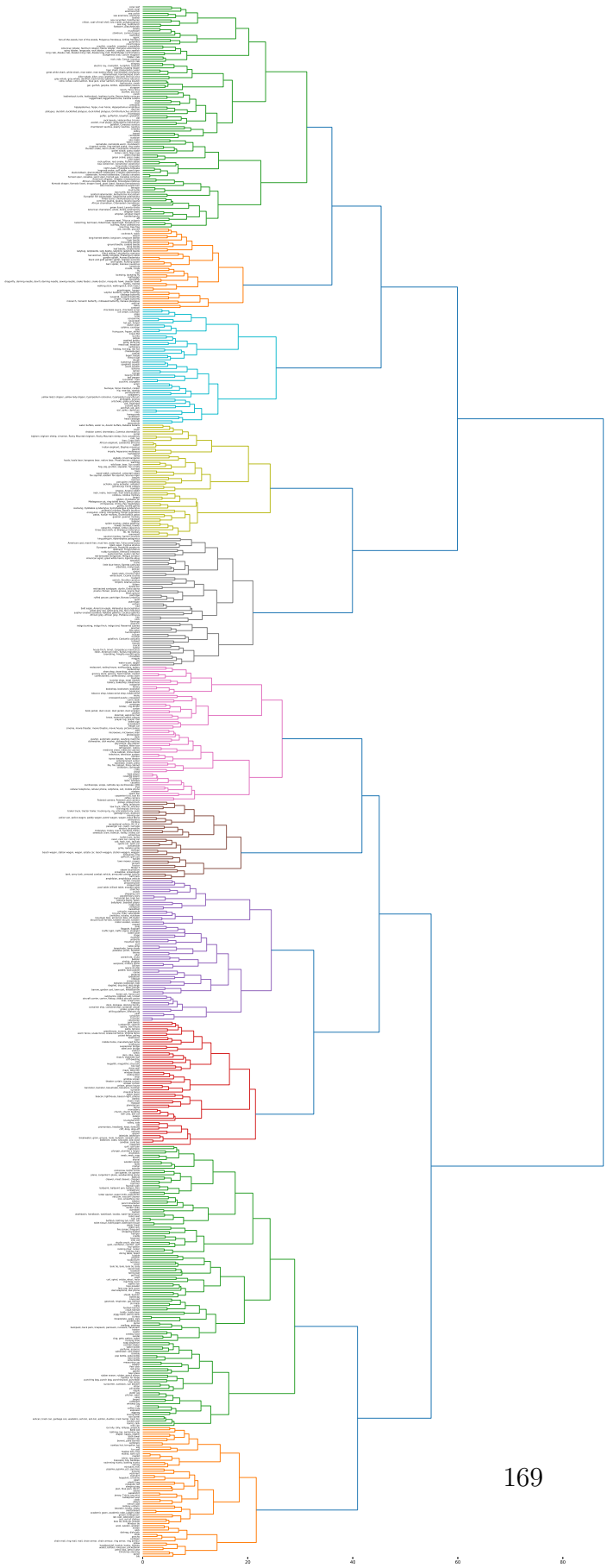


Figure 5.15 – Visual (V). dendrogram of classes built using Ward's method (CC-BY)

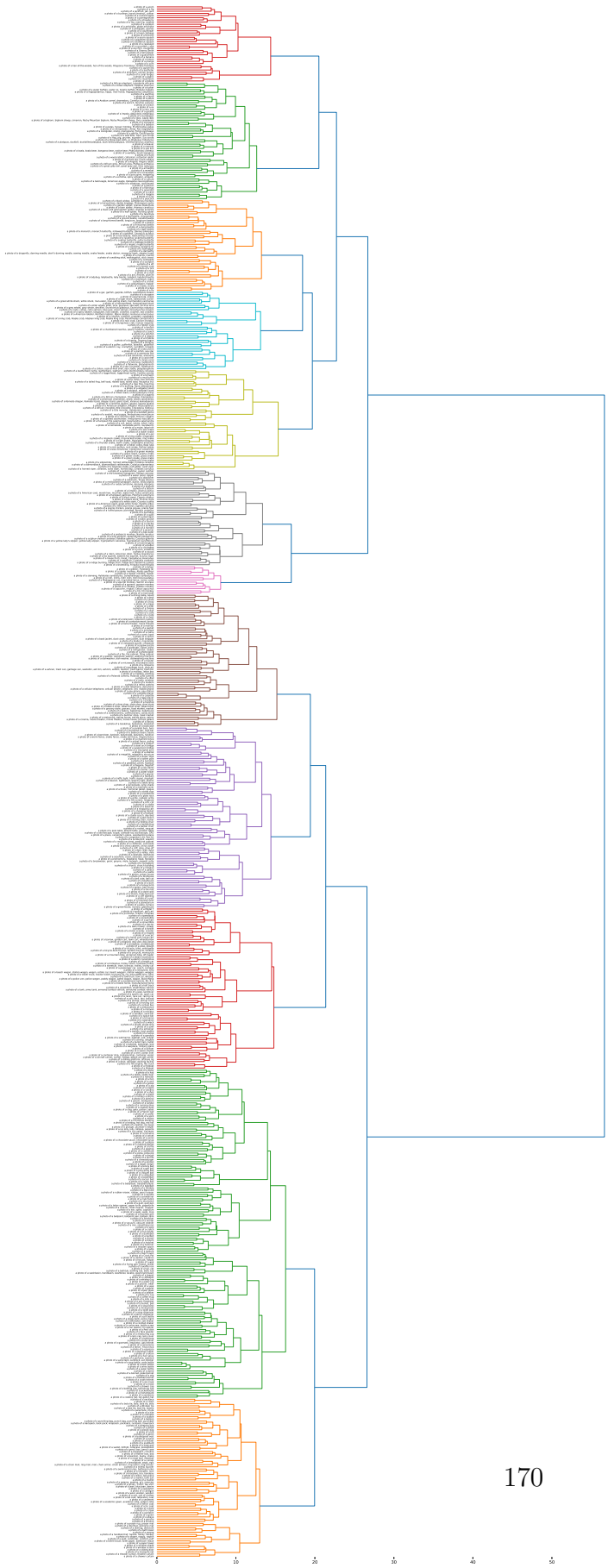


Figure 5.16 – Semantic (Se.) dendrogram of classes built using Ward's method (CC-BY)

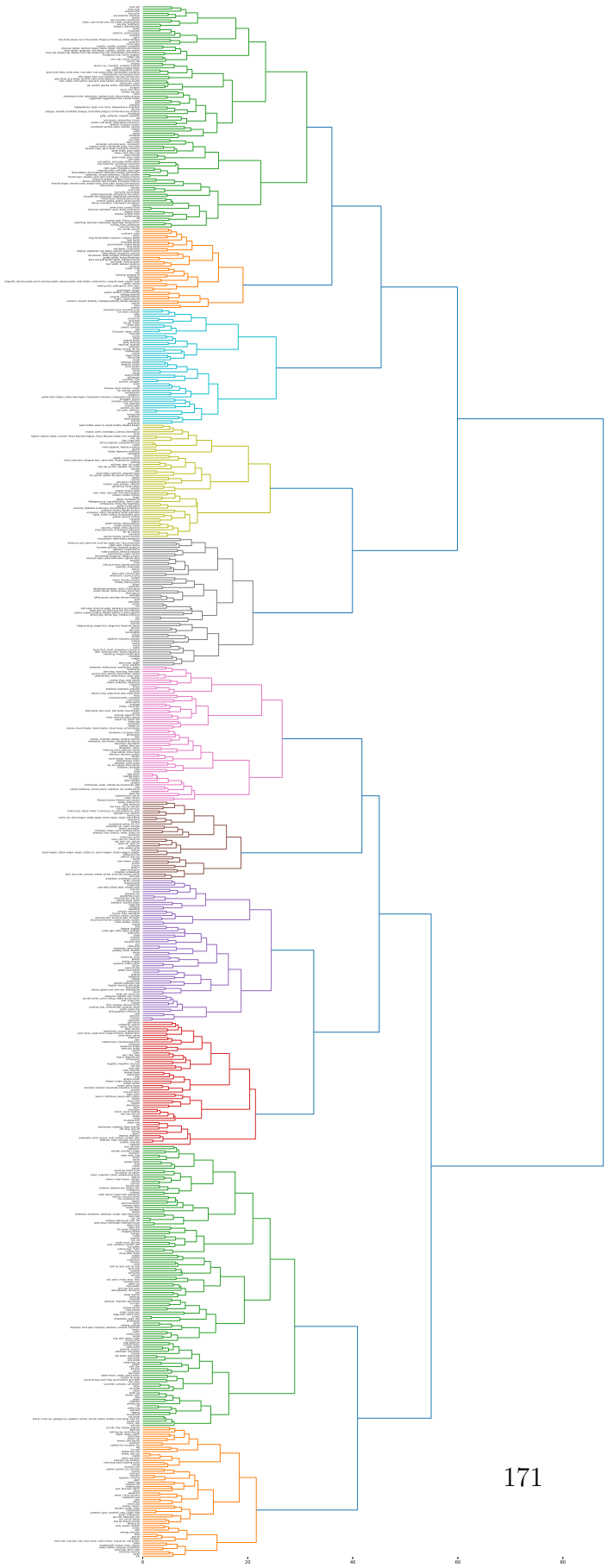


Figure 5.17 – Visual-Semantic (X). dendrogram of classes built using Ward's method (CC-BY)

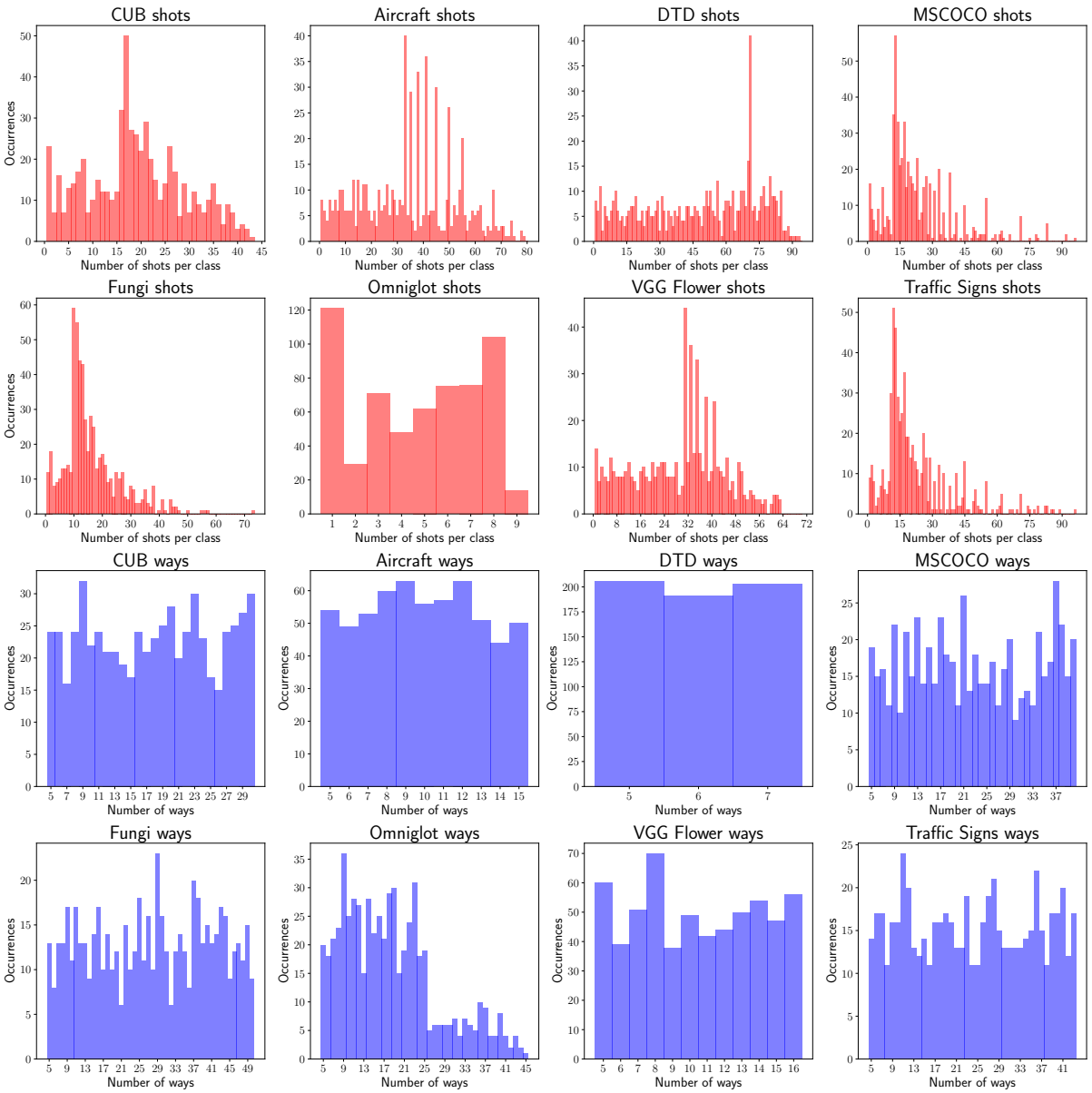


Figure 5.18 – Histogram of the number of shots and ways for each dataset using MD sampling. This shows the great variability of the sampling procedure described in [8]. (CC-BY)

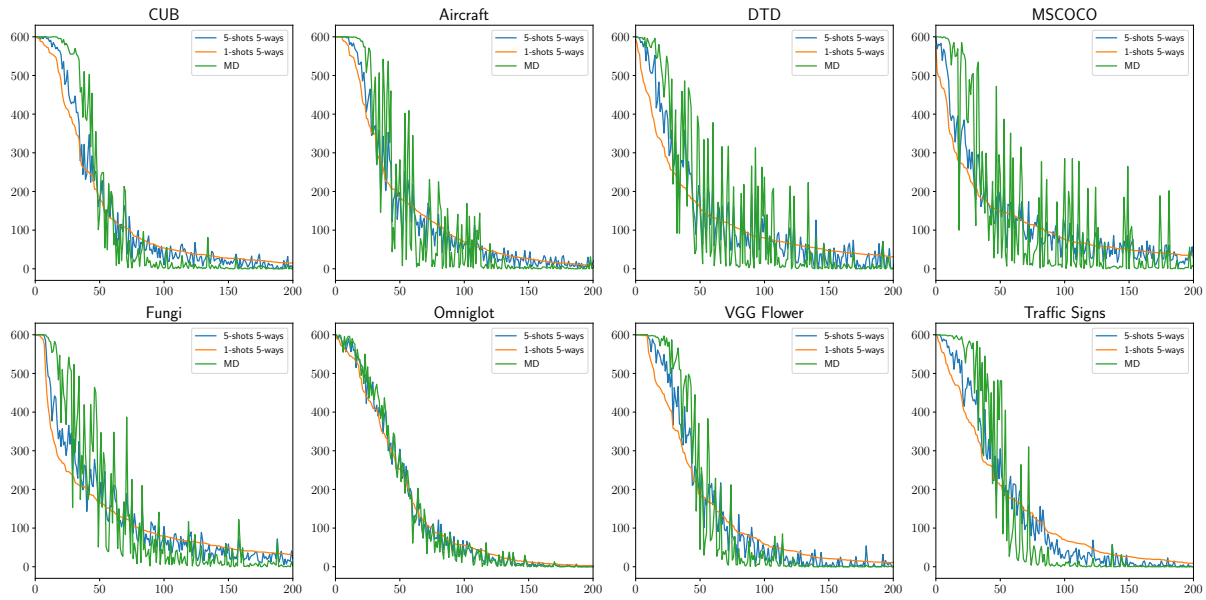


Figure 5.19 – Number of selections of ImageNet1k classes. The classes are ordered to be less and less selected in 1-shots 5-ways. We observe a strong difference in class selection between samplings. 1-shot 5-ways is clearly less consistent across episodes since the initial plateau depicting the base classes which are selected in all 600 episodes (top left of each plot) is often much smaller (if present at all) for these tasks. (CC-BY)

COLLEGES SCIENCES
BRETAGNE POUR L'INGENIEUR
LOIRE ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THE UNIVERSITY of ADELAIDE

Titre : Approches centrées sur les données pour l'adaptation dans l'apprentissage parcimonieux.

Mot clés : Apprentissage Parcimonieux, Apprentissage Profond, Vision par ordinateur, Intervalle de Confiance

Résumé : Cette thèse présente trois contributions principales visant à faire progresser l'apprentissage par peu d'exemples (ou parcimonieux) (Few-Shot Learning, FSL) en améliorant la robustesse des modèles, l'évaluation précise des performances, et l'adaptation spécifique aux tâches. Tout d'abord, nous explorons des méthodes pour construire des extracteurs de caractéristiques robustes en intégrant des ingrédients simples, ce qui permet d'atteindre des performances à l'état de l'art dans des tâches de classification en domaine. Ensuite, nous abordons la nécessité d'évaluations fiables des méthodes FSL en mettant l'accent sur les intervalles de confiance, révélant que les approches d'évaluation prédominantes négligent souvent l'aléa des données, entraînant des conclusions spécifiques à certains jeux de données. Nous proposons des techniques d'évaluation qui tiennent compte de cette variabilité, démontrant que les revendications de supériorité entre méthodes peuvent changer en conséquence voire s'inverser. Enfin, nous introduisons une approche centrée sur les données, améliorant l'adaptation aux tâches inter-domaines en oubliant sélectivement certaines portions du jeu de données de pré-entraînement, ce qui permet de réallouer l'espace des caractéristiques pour améliorer la généralisation. Ensemble, ces contributions offrent des perspectives complètes pour le développement de modèles FSL robustes et adaptatifs.

Title: Few-Shot Learning: A Data-centric Approach for Adaptation

Keywords: Few-shot Learning, Deep Learning, Computer Vision, Confidence Intervals

Abstract: This thesis presents three key contributions aimed at advancing Few-Shot Learning (FSL) through improved model robustness, accurate performance assessment, and task-specific adaptation. First, we explore methods for building robust feature extractors by incorporating simple ingredients, achieving state-of-the-art performance in in-domain classification tasks. Next, we address the need for reliable evaluations of FSL methods by showing that the predominent evaluation protocol is misleading in that it does not account for the randomness of the data, leading to conclusions that may be dataset-specific. We propose evaluation techniques that account for this randomness, demonstrating that claims of superiority between methods can change under these considerations. Lastly, we introduce a data-centric approach that enhances cross-domain task adaptation by selectively forgetting portions of the pre-training dataset, reallocating feature space to improve generalization. Together, these contributions provide comprehensive insights for developing robust, adaptable FSL models.