Més enllà de l'Aprenentatge Supervisat: Revisió de Models Agnòstics de Meta-Learning

David Cordón Sarabia*

7 de juny de 2022

*Tutor: Jordi Gonzàlez Sabaté, Dept. de Ciències de la Computació, UAB

Resum

En aquest treball explorarem els conceptes de Meta-Learning i Few-Shot Learning, així com la literatura que hi ha al respecte i sobre com resoldre aquest tipus de problemes. Explorarem els principals algorismes que els resolen: Matching Networks [1], Prototypical Networks [2], Relation Networks [3] i MAML [4], centrant-nos especialment en aquest últim. Investigarem quina és l'eficàcia de MAML en diferents problemes de regressió i classificació, i la compararem amb els altres tres mètodes mencionats anteriorment. Per últim, repassarem els problemes que MAML presenta i quines són les seves possibles millores, per així poder entendre perquè aquest algorisme ha estat tan important per a la comunitat de l'aprenentatge computacional i quins serien els següents passos en el seu desenvolupament.

Abstract

In this project we will explore the concepts of Meta-Learning and Few-Shot Learning, as well as the literature about it and how to solve these types of problems. We will explore the main algorithms that solve them: Matching Networks [1], Prototypical Networks [2], Relation Networks [3] and MAML [4], focusing on this last method. We will investigate the performance of MAML in different problems of regression and classification, and we will compare it with the three other algorithms mentioned above. At the end, we will explain the problems that MAML presents and what are their possible improvements, to be able to understand why this algorithm has been so important in the machine learning community and which will be the following steps in its development.

Índex

1 Introducció

L'objectiu principal d'aquest treball és explorar els usos del Meta-Learning, centrant-nos en el model desenvolupat en [4], anomenat *Model-Agnostic Meta-Learning* (MAML). Explorarem aquest algorisme, un dels més importants en el camp del Meta-Learning [5], per tal de poder entendre i respondre diferents preguntes com: què és el Meta-Learning i per què serveix? per què aquest model ha estat tan important per a la comunitat de l'Aprenentatge Computacional? què fa que funcioni tan bé? i finalment quins defectes o problemes presenta i, per tant, com es poden millorar?

Per tal de donar respostes a aquestes preguntes, ens centrarem en altres treballs molt importants en el camp del Meta-Learning, per entendre (i) com utilitzar el funcionament de MAML, (ii) com està descrita la seva base teòrica i matemàtica per poder conèixer les passes d'aquest algorisme, (iii) com s'analitza el seu rendiment, recreant els mateixos experiments realitzats pels autors, i (iv) com examinar els resultats que s'obtenen per poder-ho comparar amb els últims avenços en aquest camp, i justificar els seus punts forts i febles. Aquests punts febles seran la base per poder millorar el rendiment d'aquest mètode.

1.1 Què és el Meta-Learning?

L'objectiu d'aquest camp de l'aprenentatge computacional és entendre com poder resoldre problemes d'aprenentatge amb major flexibilitat, entesa com la utilització del menor nombre de mostres necessari per realitzar el màxim nombre de tasques possible [6]. Per tal de fer això, els mètodes més utilitzats apliquen un processament posterior a les dades resultants d'aplicar diferents algorismes d'aprenentatge [7]. Dit d'una altra manera, el Meta-Learning és el procés que busca aprendre a aprendre, on a través dels resultats (o meta dades) obtinguts de diferents algorismes intenta crear o millorar un nou algorisme per fer-lo més general i flexible [4].

1.2 Few-Shot Learning

Podem definir el Few-Shot Learning com una subàrea del Meta-Learning que consisteix a classificar moltes mostres havent modelitzat unes poques mostres d'entrenament [8]. Així, quan es parla d'un problema de classificació en Few-Shot Learning ens referim a ell com *N-Way K-Shot classification* [8], on N és el nombre de classes i K és el nombre de mostres per classe que utilitzarem com a conjunt d'entrenament. El valor corresponent a la K normalment varia entre 0 i 10 [9], ja que amb tan poques mostres per classe és impossible aconseguir uns bons resultats utilitzant un model estàndard. Per tal de poder fer una bona classificació havent vist prèviament tan pocs (o inclús cap) exemples d'aquesta classe, es necessita tenir en compte certa experiència prèvia, sent la modelització d'aquesta experiència la part més important del Few-Shot Learning [6]. És a la modelització d'aquesta experiència prèvia on entra el Multi-Task Meta-Learning.

1.3 Multi-Task Meta-Learning

Resoldre un únic problema de Few-Shot és complicat, ara bé, si tinguéssim diversos problemes de Few-Shot, podríem transmetre i aplicar la informació apresa d'un problema (o tasca) a un altre. El Multi-Task Meta-Learning és justament això, agrupar un conjunt de tasques per resoldre-les conjuntament. En el nostre cas, totes les tasques que agruparem seran problemes de Few-Shot Learning.

Per tant, el Multi-Task Learning és l'aprenentatge sobre diverses tasques. Un exemple de Multi-Task Learning pot ser, donada una imatge d'una cara tenim diverses tasques a realitzar (dir quin és el seu color de cabell, el seu color d'ulls, si és un home o una dona,...). D'aquesta manera podem definir el Multi-Task Meta-Learning, on partim d'una distribució de diverses tasques i un model per a cada tasca. A partir d'aquests models i els resultats que obtenim la informació necessària per poder inferir un nou model per a una tasca nova. D'aquesta manera fem Multi-Task Learning, fent un model per a cada tasca i entrenant-los. Un cop tenim els resultats del seu entrenament, podem utilitzar-los per fer Meta-Learning.

Aleshores, si tenim el nombre suficient de tasques, podem transmetre una quantitat considerable de coneixement i experiència, per tal de fer un model que sàpiga aprendre. Això implica que aquest nou model només haurà d'especialitzar-se en la nova tasca que vulgui aprendre, ja que posseeix coneixements generals sobre l'aprenentatge, adquirits gràcies a les meta-dades dels altres algorismes. Per tal d'especialitzar-se en la nova tasca no necessitarà grans quantitats de mostres, sinó que amb només unes poques mostres en tindrà suficient. Aquest aprenentatge amb poques mostres d'entrenament és el que coneixem com a Few-Shot Learning.

1.4 Estructura d'aquest treball

En aquest treball estudiarem les diferents aproximacions per resoldre el problema del Meta-Learning, però principalment ens centrarem en el Model-Agnòstic Meta-Learning [4]. La resta del treball doncs s'organitza de la següent manera: en la propera secció repassarem l'estat de l'art en Multi-Task Meta-Learning; després, entrarem en detall en l'algorisme Model-Agnòstic Meta-Learning amb el que s'han obtingut resultats molt prometedors, ja que per tal de poder fer un bon anàlisis de l'algorisme i poder identificar els seus punts forts i febles, primer hem de comprendre com funciona; analitzarem doncs aquest algorisme en dues bases de dades en la següent secció per trobar els avantatges i inconvenients, i els resultats ens portaran a les conclusions i vies futures de continuació, que tanquen aquest treball.

2 Estat de l'Art

Un cop arribats a aquest punt i plantejat el problema, la pregunta que ens ve al cap és: Com implementem un algorisme capaç de transmetre el coneixement i l'experiència obtinguda d'una tasca a l'altra? Aquest algorisme no només ha de ser capaç de transmetre el coneixement entre tasques, sinó que a més ha d'aplicar-lo a poder aprendre noves tasques amb poques mostres d'entrenament, és a dir, ha de poder resoldre el problema del Few-Shot Learning i del Meta-Learning.

En aquest camp tan recent podem destacar quatre treballs principals basats en xarxes neuronals: Matching Networks, Prototypical Networks, Relation Networks i Model Agnostic Meta-Learning. Aquests treballs són els primers que van sortir, els més referenciats de la literatura, els que han aconseguit uns millors resultats en el problema del Few-Shot Meta-Learning, i a partir dels quals han sorgit molts treballs posteriors que els milloren.

Les Matching Networks [1] van ser desenvolupades el 2016 per un equip de Google DeepMind. La inspiració dels autors va venir quan van voler ajuntar els avantatges que oferien els algorismes no paramètrics amb la potència de les xarxes neuronals, les quals són models paramètrics. Un model paramètric és aquell que la seva sortida és causada per una funció ben definida, com per exemple un polinomi on els paràmetres han de ser estimats. Alguns exemples de models paramètrics poden ser la regressió polinomial o la regressió logística. Un model no paramètric és aquell el qual els seus resultats de sortida possibles no estan parametritzats per una funció concreta, com pot ser un model basat en K-Nearest Neighbors.

Aquest algorisme es basa doncs en etiquetar cada mostra com una combinació lineal de la distància-ponderada entre les etiquetes reals de cada mostra. D'aquesta manera, la probabilitat que una mostra x^* pertanyi a una classe k ve donada per:

$$p(y^* = k|x^*, S) = \sum_{i=1}^{|S|} \alpha(x^*, x_i) 1_{y_i=k} \tag{1}$$

on y^* és la classe real de la mostra x^*, el valor k és cadascuna de les classes possibles, S és el nostre *support set*, 1_A és la funció característica (que en aquest cas particular valdrà 1 si $y_i = k$ i 0 en qualsevol altre cas) i $\alpha(x^*, x_i)$ és la similitud cosinus entre el model paramètric $g(x^*)$ i $g(x_i)$, on la funció g és la funció que porta cada mostra a l'embedding space en el qual estem treballant.

Aquest algorisme pot ser interpretat com un classificador KNN (k-nearest-neighbors) ponderat [2] aplicat a un embedding space. A la Figura 1 podem veure un esquema del funcionament de l'algorisme. Aquest algorisme ha sigut aplicat a diversos problemes pràctics de Few-Shot Learning com per exemple al reconeixement d'activitats humanes [10].

Les Prototypical Networks [2] van ser desenvolupades per Jake Snell, Kevin Swersky i Richard S. Zemel. La idea que hi ha darrere d'aquest algorisme és aprendre un espai mètric on la classificació pot ser feta només calculant distàncies entre cada mostra i un prototip representatiu de cada classe. Amb

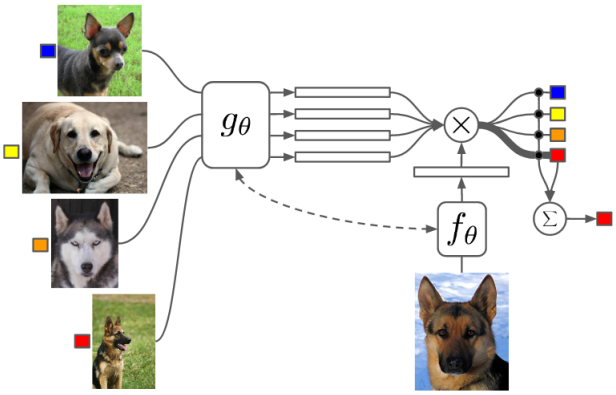

Figura 1: Esquema de l'algorisme utilitzat a una Matching Network. Figura extreta de l'article [1].

aquest algorisme aconsegueixen aplicar-ho a Zero-Shot Learning, on aconsegueixen predir la classe d'una nova mostra sense haver vist abans cap mostra d'aquella classe, només tenint una descripció semàntica de com han de ser les mostres pertanyents a aquesta classe.

Aquest algorisme doncs construeix un prototip per a cada classe i es classifica cada mostra com la classe de la qual el prototip es troba 'més proper' segons la distància Euclídea. Més concretament, la probabilitat que una mostra x^* sigui d'una classe k és:

$$p(y^* = k|x^*, S) = \frac{exp(-||g(x^*) - c_k||_2^2)}{\sum_{k' \in \{1,...N\}} exp(-||g(x^*) - c'_k||_2^2)} \tag{2}$$

on y^* és la classe real de la mostra x^*, el valor k representa cadascuna de les classes possibles, S és el nostre *Support set*, g és la funció que porta cada mostra a l'embedding space en el qual estem treballant i c_k és el prototip per la classe k.

Les Relation Networks [3] van ser desenvolupades el 2018 per un equip d'investigadors provinent de la Queen Mary University of London, la University of Oxford i de la University of Edinburgh. Aquest algorisme es basa en aplicar una embedding function a les imatges que sabem de quina classe són i a les que no ho sabem, per tal de tenir una representació de les dues imatges amb la que poder treballar. Un cop fet això comparem les dues representacions per tal de veure si coincideixen. Si hi ha una certa similitud entre elles, podem classificar-les com a imatges de la mateixa classe.

Les Relation Networks estan formades per una funció embedding g i un 'mòdul de relació' parametritzat per diferents capes d'una xarxa neuronal addicional. Apliquen g a cada query i a cada mostra del support set per poder tenir cadascun d'aquests elements representats en un embedding space. Es crea un prototip p_c per a cada classe c fent la mitjana dels embeddings del seu support

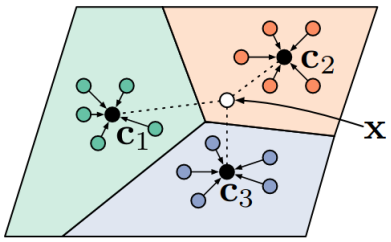

Figura 2: Esquema de l'algorisme utilitzat a una Prototypical Network. Figura extreta de l'article [2].

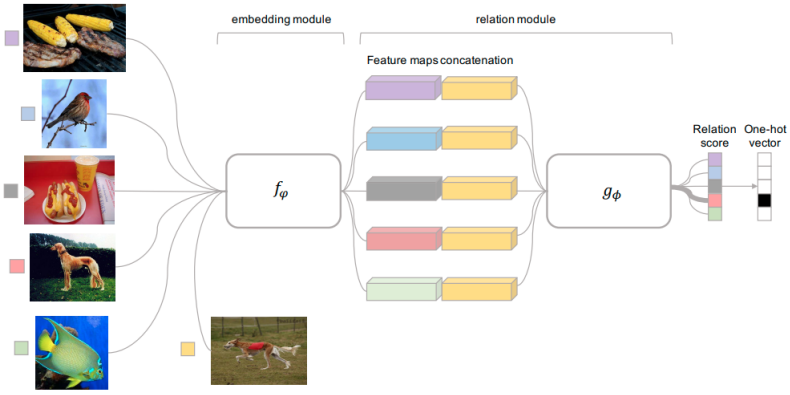

Figura 3: Esquema de l'algorisme utilitzat a les Relation Network. Figura extreta de l'article [3].

set. Es concatena cada query amb cada prototip p_c i es passa a través del mòdul relacional. El resultat que obtenim d'haver passat cada query concatenada amb el mòdul p_c és la probabilitat que aquella query pertanyi a la classe c. La loss és el *mean square error* de la predicció comparada amb el ground truth (el qual és binari). Tant g com el mòdul relacional són entrenats per minimitzar aquesta loss. A la Figura 3 podem veure un esquema del funcionament de l'algorisme.

I finalment, l'algorisme Model Agnostic Meta-Learning [4] va ser desenvolupat per Chelsea Finn, Pieter Abbeel i Sergey Levine. Just un any després de la publicació de l'article [1] on es va desenvolupar l'algorisme de les Matching Networks, i el mateix any en el qual es va presentar les Prototypical Networks. Aquest equip va voler fer un model més senzill. A més, les Matching Networks i les Prototypical Networks que eren els models que hi havien fins el moment, són models només aplicables a la classificació (com a mínim directament aplicables). La idea de MAML era ser un algorisme senzill i aplicable a diversos tipus de

tasques, independentment de si l'objectiu és classificació, regressió o *reinforcement learning*. És per això que MAML és un algorisme directament aplicable a qualsevol mètode que implementi un descens de gradient, la qual cosa implica particularment qualsevol model que implementi una xarxa neuronal estàndard. La idea que hi ha darrere d'aquest algorisme és intentar trobar uns paràmetres generals sobre el conjunt de tasques que estem utilitzant per fer Meta-Learning. Aquests paràmetres generals han de tenir la particularitat que, al cap de poques iteracions de descens de gradient sobre cada tasca individual, són capaços d'aconseguir uns bons resultats. A més és un dels algorismes més referenciats i utilitzats per resoldre problemes de Meta-Learning i Few-Shot Learning en articles com [11] o [12]. És per això que hem escollit aquest algorisme per a desenvolupar-lo en aquest treball.

3 Metodologia Model Agnostic Meta-Learning

Com s'ha comentat anteriorment, el Model Agnostic Meta-Learning és un algorisme que intenta ser general i agnòstic respecte el tipus de tasca que es vol resoldre, a més de directament aplicable a qualsevol model que es basi en un descens de gradient. La idea bàsica que hi ha al darrere d'aquest algorisme és trobar una bona inicialització dels paràmetres del model que s'utilitzen en una optimització de descens de gradient.

Per tal de fer això es seleccionen un conjunt de tasques T, i per cada tasca $t_i \in T$ realitzem l'optimització del model per reduir el valor de la funció de cost respecte a cada tasca concreta. Per exemple, volem classificar diferents dibuixos fets a mà de diferents caràcters de l'alfabet tibetà. D'aquest alfabet tenim poques mostres de cadascun dels seus caràcters, però disposem de diferents mostres de l'alfabet armeni, del coreà i del grec. En aquest cas farem una primera optimització del model per classificar els caràcters en l'alfabet armeni, una segona optimització pel model que classifica els caràcters de l'alfabet coreà i una tercera del model de l'alfabet grec. Un cop estan tots els paràmetres de cada tasca optimitzats, es fa una darrera optimització general (anomenat *meta-descens de gradient* [4], ja que es fa sobre les meta-dades obtingudes com a resultats dels models corresponents a cada tasca). Aquest procés es fa per tal de poder trobar els millors paràmetres d'un model general aplicable a totes les tasques. La idea principal serà aconseguir una bona inicialització de paràmetres per aconseguir uns bons resultats per a cada tasca.

3.1 L'Algorisme MAML

L'objectiu principal de MAML és que aprengui una inicialització dels paràmetres necessaris d'una manera que estiguin preparats per una ràpida adaptació a una nova tasca. Aquest algorisme no fa cap assumpció respecte de la forma del model, excepte que està parametritzat a través d'un vector de paràmetres θ, i que la funció de loss és prou derivable.

Algorithm 1 Pseudocode of MAML

1: Require: p(T) distribution over tasks
2: Require: α, β step size hyperparameters
3: randomly initialize θ
4: **while** not done **do**
5: sample batch of tasks $\mathrm{T}_i \sim$p(t)
6: **for** all T_i **do**
7: Evaluate $\nabla_\theta L_{T_i}(\mathrm{f}_\theta)$ with respect to K examples
8: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta)$
9: **end for**
10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_{i \sim p(T)}} L_{T_i}(f_{\theta_i'})$
11: **end while**

Si ens fixem en l'Algorisme 1 veiem que entre les línies 1 i 3 que només necessitem escollir una distribució de tasques, dos hiperparàmetres (els dos learning rates) i un vector de paràmetres. A continuació, per cada tasca optimitzem els paràmetres, especialitzant-los en cadascuna d'elles, obtenint θ_i' per a cada tasca T_i. Quan hem acabat d'optimitzar els paràmetres θ per a cada tasca, calculem la funció de loss sobre cada tasca amb els paràmetres actualitzats θ_i' i utilitzem aquests resultats per actualitzar els paràmetres inicials θ tal com podem veure a la línia 10. Podem veure que els passos 7 i 8 es poden repetir tants cops com actualitzacions del gradient vulguem per a cada tasca.

Aquest algorisme bàsicament el que proposa és que, un cop fet el descens de gradient per a cada tasca i tinguem tots els seus paràmetres actualitzats, actualitzem la inicialització de pesos que tenim en la direcció de màxim decreixement que ens portarà a minimitzar la suma de les losses de totes les tasques. Amb això aconseguim uns paràmetres actualitzats pels quals la suma de les funcions loss de totes les tasques serà més petita (després d'haver actualitzat els paràmetres específicament per a cada tasca). Com que la funció loss depèn d'un vector θ de n paràmetres, podem expressar la funció de loss com $L(\theta)$, la qual ens representa una funció en la dimensió $n+1$. D'aquesta manera, podem pensar que la cerca del vector de paràmetres θ és la cerca d'un punt tal que la seva representació sobre totes les funcions de loss de cada tasca estigui prop d'un mínim, al qual podem arribar en poques iteracions de descens de gradient per tal d'assolir el mínim de la tasca.

Podem veure gràficament a la Figura 4 com els paràmetres θ que s'han buscat i optimitzat amb un descens de gradient tenen la propietat que poden adaptar-se a cadascuna de les tasques amb poques (inclús una) iteracions de descens de gradient sobre aquella tasca. Ja podem observar que un possible inconvenient d'aquest algorisme és que quan fem l'actualització dels paràmetres a la línia 10, estem calculant el gradient sobre uns paràmetres ja actualitzats amb descens de gradient, el que implica calcular un gradient sobre un gradient, la qual cosa és computacionalment costosa.

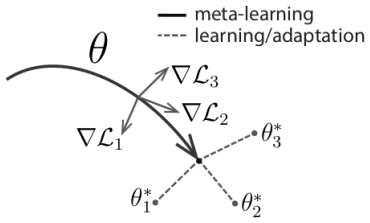

Figura 4: Vector de paràmetres θ optimitzat per tal de poder assolir el mínim de la funció loss de cada tasca després de poques (o una) iteracions en la tasca T_i per obtenir θ_i^*. Figura extreta de l'article [4].

3.2 Aplicacions del MAML

Com hem comentat anteriorment, aquest algorisme és agnòstic, en el sentit que és independent de l'arquitectura de la xarxa utilitzada o de la distribució de tasques. És per això que el mètode és aplicable tant a *supervised learning* (incloent-hi regressió i classificació) com a *reinforcement learning*. En aquest treball ens centrarem en la implementació de MAML amb l'objectiu de regressió i, principalment, classificació.

3.2.1 Classificació

Si el nostre model és entrenat per fer classificació, la funció que utilitzarem com a funció loss serà la Cross-Entropy Loss, definida com:

$$L_{T_i}(f_\theta) = \sum_{x^{(j)},y^{(j)}\sim T_i} y^{(j)} log(f_\theta(x^{(j)})) + (1-y^{(j)}) log(1-f_\theta(x^{(j)})) \qquad (3)$$

on $(x^{(j)}, y^{(j)})$ són les parelles input/output seleccionades de la tasca T_i. Podem definir un problema de *N-Way K-Shot classification* com un problema de classificació amb N classes i K parelles input/output per classe, el que fa un total de NK mostres.

Els autors [4] especifiquen al seu article quina és l'arquitectura de la xarxa utilitzada en aquest cas. La xarxa neuronal es basa en 4 mòduls amb una convolució 3x3 i 64 filtres, seguit de *batch normalization*, no-linearitats ReLU i un max-pooling 2x2. L'última capa és passada per un soft-max. Aquesta arquitectura base pateix algunes petites modificacions en funció de quin del dos datasets que veurem després és utilitzat. Pel primer dels datasets, l'Omniglot [13], en comptes d'utilitzar un max-pooling, s'utilitzen *strided convolutions*. Pel segon dels datasets, el Mini-ImageNet, s'utilitzen 32 filtres en comptes de 64, per tal de reduir l'overfitting. Els detalls del rendiment en aquestes bases de dades es veuran en la secció següent.

Un cop hem entès el problema de classificació que volem resoldre, replantejarem l'algorisme basat en l'Algorisme 1 centrant-nos en la creació dels conjunts d'entrenament, a més de veure més a fons la validació:

Algorithm 2 Validation of MAML

1: **for** each task T_i **do**
2: Choose N random classes
3: **for** each class **do**
4: Choose 2k random samples and divide them in *Set A* and *Set B* with the same number of elements
5: **end for**
6: **end for**
7: **for** each task T_i **do**
8: Use *Set A* of all classes to perform gradient descent over θ to obtain $\theta_i' = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta)$
9: Use *Set B* of all classes to compute the training loss for this task using θ_i'
10: **end for**
11: Total loss: Mean(All tasks losses from set B)
12: Total accuracy: Mean(All tasks accuracies from set B)
13: (only if we are in the training phase) Update the initial parameters θ using an optimizer and the total loss with θ: $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta_i'})$

Tal com podem veure entre els passos 1 i 4 de l'Algorisme 2, MAML consta de les mateixes classes i de les mateixes imatges per classe durant la fase d'entrenament i la fase de validació, a diferència d'alguns dels altres algorismes que hem vist com ara les Matching Networks [1] o les Prototypical Networks [2].

3.2.2 Regressió

Per un model entrenat per fer regressió amb MAML, la funció loss utilitzada serà la Mean-squared Error (MSE), definida com:

$$L_{T_i}(f_\theta) = \sum_{x^{(j)}, y^{(j)} \sim T_i} ||f_\theta(x^{(j)}) - y^{(j)}||_2^2 \tag{4}$$

on, de la mateixa manera que a un problema de classificació, $(x^{(j)}, y^{(j)})$ són les parelles input/output seleccionades de la tasca T_i. Podem definir un problema de *K-shot regression* com un problema de regressió on únicament són proporcionats K parelles input/output.

En aquest cas, els autors de MAML [4] també especifiquen quina arquitectura de la xarxa han utilitzat. Pel model de regressió s'ha utilitzat una xarxa neuronal amb 2 capes ocultes de tamany 40 amb una funció d'activació ReLU que ens eliminen els inputs negatius, enviant-los tots a 0.

3.3 Aspectes Pràctics del MAML

Per una banda, tot i la gran eficiència del Model-Agnostic Meta-Learning, aquest dista de ser un algorisme perfecte. Presenta diferents problemes i defectes molt

diversos, els quals van des d'inestabilitats durant l'entrenament del model a causa de la inicialització de diversos paràmetres o hiperparàmetres fins a un gran cost computacional en el càlcul de les segones derivades de la funció a optimitzar, entre d'altres [14]. Resolent aquests problemes doncs aconseguirem un model més eficient i ràpid, al mateix temps que amb una capacitat de generalització dels paràmetres més elevada, la qual disminuirà les iteracions necessàries per a l'optimització d'aquests. A més, a l'augmentar l'estabilitat del model, l'eficàcia d'aquest serà independent dels paràmetres inicials i, per tant, el model serà capaç de generalitzar millor. A més, amb una bona generalització també augmentarem l'eficàcia del model. Tot això ho analitzarem en la següent secció.

D'altra banda, en funció de quin sigui el nostre objectiu i problema a solucionar, aquesta cerca d'uns paràmetres fàcilment adaptables pot tenir diverses interpretacions, depenent de si s'aplica com un mètode d'inicialització de paràmetres en un problema de múltiples tasques, o com un mètode per seleccionar les millors característiques per a resoldre l'aprenentatge.

Si el nostre objectiu és resoldre un problema de Few-Shot Learning, utilitzarem el MAML com un mètode d'inicialització dels paràmetres, que utilitzarem per fer descens de gradient sobre el nostre conjunt de NK mostres, amb només K mostres per cada classe.

Per altra banda, si el nostre objectiu és resoldre un problema de Meta-Learning, utilitzarem el MAML com un mètode d'optimització d'uns paràmetres els quals es fixaran en les característiques concretes (les millors i més significatives) que ens donen més informació per a un grup de tasques específic. Per tal de veure aquesta selecció de les millors característiques podem mirar quines són les sortides de les diferents capes convolucionals de la xarxa. D'aquesta manera, mirant aquestes sortides ens podrem fer una idea de quines són les característiques en les quals ens estem fixant.

A la Figura 5 podem observar diferents resultats de diverses capes convolucionals. A la primera fila de la imatge, podem interpretar que la xarxa el que està buscant són contrastos de la imatge, per poder definir bé quina és la frontera del dibuix i poder diferenciar millor la forma. A la segona fila sembla que la xarxa està dilatant la silueta i el traç del dibuix. Per últim, a la tercera i quarta fila és més difícil poder interpretar que està fent aquí la xarxa. Podem teoritzar que el que està fent és trobar contrastos a la imatge, intentant polaritzar les zones clares cap a blanc i les zones més fosques cap al negre.

4 Resultats Experimentals

El model MAML implementa una bona cerca de paràmetres generals i, per tant, és directament aplicable a qualsevol algorisme que utilitzi un descens de gradient, ja sigui amb l'objectiu de fer classificació o de fer regressió. Això fa que MAML sigui molt versàtil, a més de senzill, ja que no utilitza una quantitat excessiva d'hiperparàmetres. És simple, general i aconsegueix uns resultats que igualen l'estat de l'art [15], com es mostra en aquesta secció.

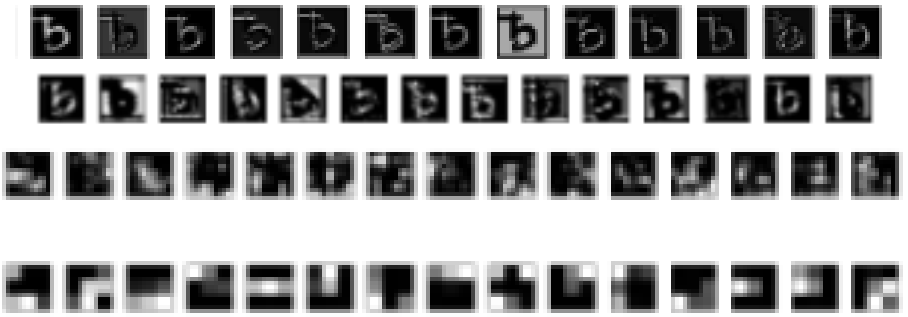

Figura 5: Sortides de diferents capes convolucionals d'un model ja entrenat amb Omniglot.

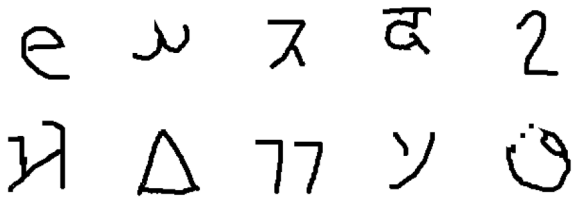

Figura 6: Exemples d'imatges de l'Omniglot dataset.

Figura 7: Exemples d'imatges pertanyents a diverses classes del Mini-ImageNet dataset.

4.1 Datasets utilitzats

Per tal de poder saber si aquest algorisme aconsegueix posar solució al problema del *Meta-Learning* i de *Few-Shot Learning* avaluarem el nostre model

en diferents experiments utilitzant diferents datasets. Per tal d'aconseguir les respostes necessàries, utilitzarem 3 datasets diferents:

El primer dels datasets utilitzats és l'Omniglot. Aquest conjunt d'imatges va ser recopilat i utilitzat per primer cop al treball fet per *Lake et al.* [13]. Els creadors d'aquest dataset l'anomenen "l'oposat de l'MNIST", ambdós consisteixen en caràcters escrits a mà, però l'MNIST consta de només 10 caràcters (classes) i milers d'exemples per classe mentre que a Omniglot tenim més de 1600 classes i només 20 exemples per cadascuna d'elles. Aquests caràcters utilitzats per Omniglot són de més de 50 alfabets diferents, incloent-hi Bengali, Cyrillic, Avorentas, Sanskrit, Tagalog, i fins i tot alfabets sintètics utilitzats en novel·les de ciència-ficció. Per tal d'aconseguir les 20 mostres necessàries de cada classe, es va fer servir *Amazon Mechanical Turk*, una eina des de la qual es va proposar a la comunitat que utilitzava aquesta plataforma que per cada mostra 20 persones diferents fessin el seu dibuix a mà. A la Figura 6 podem veure alguns exemples de com són les imatges que podem trobar a aquest dataset.

El segon dels datasets utilitzats havia de ser un que fos un repte d'una complexitat superior cap a MAML, i aquí és on entra en joc Mini-ImageNet. El Mini-ImageNet dataset va ser proposat per primer cop per *Vinyals et al.* (2016) [1] com un benchmark que oferia el repte de la complexitat de les imatges d'ImageNet [16], però sense els recursos i la infraestructura necessària que requeria utilitzar ImageNet sencer. Tot i això, després del treball realitzat per *Vinyals et al.*, la selecció i divisió de les classes de ImageNet per tal de crear Mini-ImageNet no va ser publicada. Per tal de solucionar això, un any després vam poder veure publicada una divisió del dataset original feta per [17] (2017), que és en la que es van basar els creadors de MAML [4]. Va ser en aquest mateix paper on *Lavi i Larochelle* van introduir el concepte d'episodi d'entrenament, utilitzat posteriorment a la resta d'implementacions de Meta-Learning. Aleshores un episodi es refereix a l'entrenament del model sobre una tasca concreta. D'aquesta manera, Mini-ImageNet consta de 100 classes seleccionades aleatòriament del dataset original, amb 600 imatges per classe. A la Figura 7 podem veure alguns exemples del tipus d'imatges que trobem a Mini-ImageNet i la classe a la qual pertanyen.

Finalment, els dos datasets vists anteriorment són utilitzats per un model de classificació. Quan parlem de regressió, el que es va utilitzar com a dataset per validar va ser una generació de sinusoides. Cadascuna de les tasques és definida com una sinusoide amb una amplitud i una fase concreta. Per a cadascuna de les tasques es generen K punts aleatoris pertanyents a la sinusoide. D'aquesta manera, els punts (x, y) utilitzats durant l'entrenament i la validació del model són de la forma $y = Asin(x - \phi)$, on A és l'amplitud i ϕ és la fase. Per a cada combinació de A i ϕ tenim una tasca diferent. L'objectiu d'utilitzar aquest dataset és que el model resultant aprengui la natura ondulatòria de la funció sinus, i que, per tant, en poques optimitzacions de descens de gradient només hagi d'inferir l'amplitud i la fase.

	5-Way 1-Shot	**5-Way 5-Shot**
Omniglot	99.2 ± 0.085%	**99.9 ± 0.031%**
Mini-ImageNet	49.02 ± 0.49	**64.82 ± 0.477%**

Taula 1: Resultats obtinguts a l'hora d'executar MAML per fer classificació 5-Way 1-Shot i 5-Way 5-Shot sobre els datasets de Mini-ImageNet i Omniglot.

	2-Way	**5-Way**	**10-Way**	**15-Way**
MAML [4]	69.8%	49.02%	**33.02%**	**25.06%**
Proto Nets [2]	61.88%	30.40%	24.90%	19.80%
Matching Nets [1]	61.75%	39.60%	25.64%	19.46%
Relation Nets [3]	**76.90%**	**49.48%**	32.27%	24.57%

Taula 2: Variació de l'accuracy en funció del paràmetre N i de l'algorisme utilitzat per tal de resoldre el problema de classificació N-Way 1-Shot amb Mini-ImageNet.

4.2 Rendiments Obtinguts

Un cop tenim clar sobre quins conjunts de dades s'avaluarà el model, l'hem hagut de posar a prova amb aquests datasets. Per tal de fer això hem executat el codi en el nostre ordinador de manera local per tal de poder veure i interpretar els resultats, tal com es descriu a continuació[1].

4.2.1 Resultats de Classificació

Els resultats es poden veure a la Taula 1. Aquí podem veure els resultats obtinguts en quatre problemes de classificació diferents, dos d'ells amb el dataset Omniglot i els altres dos amb el dataset Mini-ImageNet. Podem veure que als resultats obtinguts amb Omniglot obtenim una accuracy molt elevada, del 99%. Podem apreciar que no hi ha gairebé cap diferència entre un problema de classificació 5-Way 1-Shot i un de 5-Shot, ja que l'accuracy és tan elevada que no dona marge a millora. En canvi, els resultats obtinguts amb Mini-ImageNet són més interessants. Quan augmentem de 1-Shot a 5-Shot, l'accuracy es veu incrementada considerablement. És per aquest motiu, que analitzar els resultats obtinguts amb Mini-ImageNet ens pot aportar més informació sobre l'eficàcia i la capacitat de generalitzar dels models. Per això, per la resta d'experiments que farem de problemes de classificació ens centrarem en el dataset Mini-ImageNet.

Per tal d'avaluar la metodologia MAML, definim un problema de classificació N-Way K-Shot, fixant la N i la K. El dataset utilitzar serà Mini-ImageNet. Els resultats s'han separat en dues taules diferents. A la Taula 2 es poden veure els resultats obtinguts amb els diferents algorismes en un problema de classificació N-Way 1-Shot. D'aquesta manera no només podem comparar els

[1] El codi utilitzat per tal de reproduir tots els experiments d'aquest treball es pot trobar a: `https://github.com/DavCorSar/Revision_of_MAML`

	1-Shot	5-Shot	10-Shot	15-Shot	20-Shot	25-Shot
MAML [4]	69.8%	82.11%	85.61%	85.83%	83.15%	84.67%
Proto Nets [2]	61.88%	72.88%	74.00%	79.75%	79.25%	82.5%
Matching Nets [1]	61.75%	63.75%	68.37%	69.50%	64.13%	73.25%
Relation Nets [3]	**76.90%**	**86.88%**	**88.75%**	**89.95%**	**90.08%**	**90.73%**

Taula 3: Variació de l'accuracy en funció del paràmetre K i de l'algorisme utilitzat per tal de resoldre el problema de classificació 2-Way K-Shot amb Mini-ImageNet.

diferents resultats amb diversos algorismes, sinó que a més veiem com influeix el paràmetre N en la precisió del model final.

Per altra banda, podem veure a la Taula 3 els resultats d'un problema de classificació 2-Way K-Shot amb els diferents algorismes i variant el nombre de mostres per classe. S'ha de tenir en compte que a les Prototypical Networks [2] i a les Matching Networks [1], els models són entrenats als papers originals amb més classes de les quals després disposarem per fer la validació, és a dir, el model és entrenat sobre tasques de les que disposem 20 classes amb una imatge per classe, i després la validació és feta sobre una tasca amb 5 classes i una imatge per classe, per exemple. Nosaltres, per entrenar tots els models en les mateixes condicions i poder fer una comparació justa utilitzarem les mateixes classes i les mateixes mostres per classe tant en la fase d'entrenament com en la de validació per a tots els algorismes. És per aquest motiu que els resultats que hem obtingut amb aquests dos mètodes no són comparables als resultats obtinguts pels autors als seus articles originals [2] i [1].

Un cop analitzat els resultats obtinguts per l'algorisme ens preguntem, com pot variar la precisió de la nostra classificació quan canviem el nombre de classes i de mostres per classe? Per tal de veure com afecten aquests paràmetres a l'eficàcia del model hem fet un seguit d'experiments amb diferents combinacions d'aquests paràmetres. Els resultats es poden veure a la Taula 2, on mostrem els diferents resultats variant el nombre de classes, però fixant el nombre d'imatges per classe a 1. Per altra banda, a la Taula 3 podem veure els resultats de fixar el nombre de classes a 2 i variar el nombre d'imatges per classe. Degut a limitacions de memòria de la GPU utilitzada no he pogut reproduir resultats amb valors més grans de N i K.

Per tal de poder visualitzar els resultats d'haver fixat un dels dos paràmetres (N o K) i haver deixat a l'altre lliure, hem representat gràficament l'evolució de la precisió del model a mesura que augmenta un dels dos paràmetres. Podem veure gràficament els resultats corresponents de la Taula 3 a la Figura 8. Es pot veure com les corbes pertanyents als algorismes de MAML i de les Relation Networks aconsegueixen uns resultats molt millors que en el cas de les Matching Networks o les Prototypical Networks. En el cas de MAML, com era d'esperar, l'accuracy augmenta fins a arribar al màxim assolit a les 15 mostres per classe. A partir d'aquest punt comença a baixar. Això és degut a l'overfitting, és a dir, el nostre model ha entrenat massa amb el conjunt d'entrenament, i quan

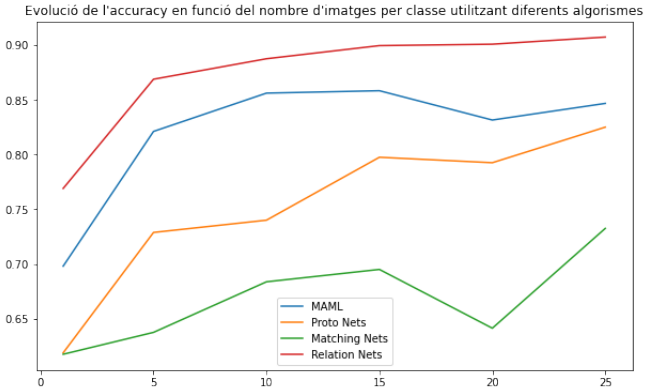

Figura 8: Variació de la precisió del model en funció del nombre de mostres per classe.

volem que generalitzi a algunes mostres diferents no sap ben bé com fer-ho. Mirant aquesta gràfica, podem concloure que MAML ha assolit el seu objectiu, generalitzar tasques senceres a partir de poques mostres d'entrenament. Per altra banda, si comparem la gràfica del creixement de MAML amb la corba de les Prototypical Network o de les Matching Network veiem que aquests dos últims algorismes continuen augmentant la seva precisió a mesura que augmenta el nombre d'imatges per classe. Això ens fa pensar que no han assolit el seu màxim d'aprenentatge, i que, per tant, l'algorisme de MAML és capaç d'aprendre més i millor amb menys imatges per classe que els altres dos algorismes, és a dir, és capaç de modelitzar l'experiència prèvia per tal de poder aplicar-la a noves tasques. Per altra banda, la corba de les Relation Networks veiem que és capaç d'aprendre i generalitzar amb poques mostres d'entrenament inclús millor que l'algorisme de MAML. A més, tot i que al final la seva corba d'aprenentatge se suavitza, al contrari que MAML no deixa d'aprendre i veiem que encara és capaç d'extreure més informació i millorar.

Per altra banda, si fixem el nombre de mostres per classe a 1 i variem el nombre de classes podem observar a la Figura 9 com el comportament de MAML i de la resta d'algorismes és l'esperat. A mesura que augmentem el nombre de classes la precisió del nostre model decreix de manera exponencial, a causa de l'augment de complexitat que suposa tenir més classes entre les quals classificar. Si comparem la corba de MAML amb la corba de les Prototypical Networks, les Matching Networks i les Relation Networks veiem que la forma i tendència que segueixen és la mateixa, la qual cosa ens indica que quan augmenta el nombre de classes en la classificació, la complexitat per generalitzar augmenta de manera proporcional amb tots els algorismes. Tot i així, podem apreciar que les corbes de les Matching i les Prototypical Networks obtenen uns resultats molt semblants entre elles, al igual que ho fan les Relation Networks amb MAML. Tot i això, veiem que quan el nombre de classes és petit (entre 2 i 5) l'eficàcia de les Relation Network supera a MAML, però a mesura que augmenta el nombre de

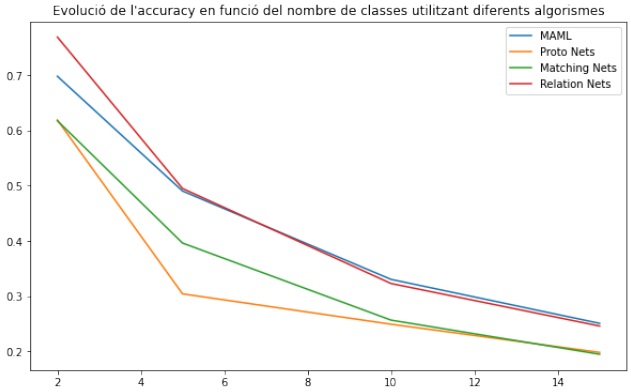

Figura 9: Variació de la precisió del model en funció del nombre de classes seleccionades.

classes aquesta diferència es fa cada cop més petita, fins que finalment sembla que MAML aconsegueix superar (per poc) a les Relation Networks.

4.2.2 Resultats de Regressió

Hem avaluat el rendiment d'un classificador entrenat amb MAML, però com hem dit al principi, un dels grans avantatges de MAML sobre la resta d'algorismes és que també és directament aplicable a la regressió. Per tal de veure els resultats obtinguts entrenarem el nostre model amb el conjunt de sinusoides descrit anteriorment. Posteriorment avaluarem el model variant dos paràmetres principalment: el nombre de punts d'entrenament (similar a la K en K-Shot classification) i el nombre d'optimitzacions de descens de gradient. Els paràmetres de l'amplitud i la fase els fixarem a dos valors aleatoris, en el nostre cas $A = 2.3$ i $\phi = 1.67$.

A la Figura 10 podem veure els resultats d'haver entrenat un model amb 1, 5, 10 i 15 mostres d'entrenament i amb 1, 5 o 10 iteracions de descens de gradient. A dalt a l'esquerra tenim els resultats d'haver entrenat un model amb només una mostra per cada sinusoide. Es pot apreciar que no hi ha cap diferència entre els resultats obtinguts amb 1 iteració de descens de gradient, 5 o 10. Això ens fa pensar que hi ha una relació entre el nombre d'iteracions i el nombre de mostres que tenim: com més gran sigui un d'aquests valors, més flexibilitat tindrem a l'altre. A la cantonada superior dreta podem veure els resultats d'haver entrenat un model amb 5 mostres per a cada sinusoide. En aquest cas sí que observem diferències entre els resultats obtinguts amb diferents iteracions de descens del gradient. Com més iteracions de descens del gradient fem, la predicció s'acosta més a la gràfica real, però sense canviar la forma inicial. A la cantonada inferior esquerra podem veure els resultats d'haver entrenat un model amb 10 mostres per a cada sinusoide. En aquest cas si ens fixem en les diferències que tenim entre els resultats podem observar que quan augmentem el nombre d'iteracions

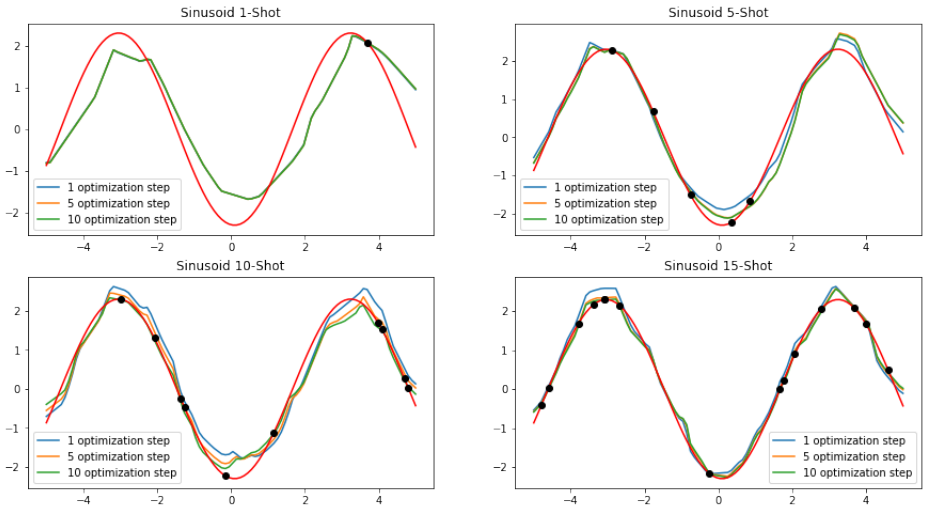

Figura 10: Resultats obtinguts entrenant una sinusoide variant el nombre de mostres d'entrenament i el d'iteracions en l'optimització.

de descens de gradient no només s'apropa a la gràfica original (com passa a la gràfica de *Sinusoid 5-Shot*) sinó que la forma de la corba se suavitza en el sentit que es redueixen les línies rectes i les cantonades i augmenten les zones corbes.

Finalment, a l'última gràfica on mostrem els resultats de la *Sinusoid 15-Shot*, podem observar com en el moment que augmentem el nombre d'iteracions de descens de gradient la corba és capaç d'adaptar-se a la natura ondulatòria de la funció sinus, suavitzant-se en la majoria dels màxims i mínims per tal de seguir la naturalesa periòdica de la funció. Tot i això, podem observar una tendència general entre totes les gràfiques, i és que per molt bé que s'adapti, les zones on li costa més és als punts de màxims i mínims.

En resum, mirant aquests resultats podem teoritzar que quan tenim poques mostres d'entrenament, el fet d'augmentar les iteracions que fem en l'optimització no provoca cap millora significativa en els resultats del model. En el moment que augmentem el nombre de mostres d'entrenament sí que podem observar diferències entre els resultats en funció de les iteracions de descens de gradient fetes. Podem teoritzar que com més mostres d'entrenament tinguem, menys iteracions necessitarem per aconseguir una diferència significativa entre resultats. Això, probablement és degut al fet que, per exemple, si realitzem 10 iteracions amb només una mostra d'entrenament haurem vist 10 punts. En canvi, si realitzem 10 iteracions amb 5 mostres d'entrenament haurem vist 50 punts. Aquesta és probablement la raó per la qual a la gràfica *Sinusoid 1-Shot* no podem apreciar cap diferència entre els resultats, mentre que a la gràfica corresponent a *Sinusoid 15-Shot* podem observar una diferència significativa entre realitzar 1 iteració de descens de gradient i 5 iteracions. En aquest últim cas, amb 5 iteracions haurem vist un total de $15 \cdot 5 = 75$ punts, mentre que als

	1 step	**5 steps**	**10 steps**
1-Shot	0.4010	0.4083	0.4083
5-Shot	0.0513	0.0811	0.0799
10-Shot	0.1107	0.0690	0.0611
15-Shot	0.0347	0.0195	0.0200

Taula 4: Variació de la funció de loss depenent del nombre de mostres d'entrenament i de la quantitat d'iteracions en l'optimització.

resultats de la gràfica *Sinusoid 1-Shot* amb 5 iteracions haurem vist només 5 punts.

5 Discussió

Com podem veure, encara que l'algorisme de MAML té una gran varietat d'avantatges respecte a altres algorismes de Meta-Learning, dista de ser un algorisme perfecte.

5.1 Problemes inherents del MAML

Un dels principals problemes que podem apreciar és que hi ha inestabilitats durant l'entrenament del model en funció de l'arquitectura utilitzada. L'objectiu de MAML és que sigui un model directament aplicable a diverses tasques, independentment del model i l'arquitectura utilitzada, però això no vol dir que sigui totalment estable amb qualsevol arquitectura utilitzada. Quan utilitzem strided convolutions en comptes de max pooling podem observar inestabilitats durant l'entrenament. [14].

L'execució de MAML comporta un cost computacional elevat amb el càlcul de segones derivades. Com hem pogut veure, aquest algorisme consisteix en optimitzar una funció loss que ja ha estat optimitzada amb un descens de gradient. Aquest fet d'aplicar una optimització sobre una optimització implica calcular segones derivades, càlcul d'un gradient sobre un gradient. Computacionalment, això té un cost elevat, i augmenta considerablement el temps d'execució del nostre programa. [14].

Les validacions de MAML és fan sobre distribucions de tasques extremadament semblants entre elles, com poden ser reconèixer diferents caràcters en Omniglot. En el moment d'avaluar els resultats obtinguts durant l'entrenament en aquesta distribució de tasques és difícil veure grans diferències, ja que com hem pogut veure els resultats sempre seran molt elevats. Per altra banda, amb Mini-ImageNet, les tasques tenen més diferències entre elles, però igualment aquesta diferència podria ser major per tal d'explotar a fons les capacitats de MAML, ajuntat diversos datasets diferents. [18].

Els learning rates estan fixats, tant del bucle intern per calcular l'objectiu per a cada tasca com el learning rate corresponent al meta-descens de gradient. Els autors de l'article [4] fixen només tres hiperparàmetres referents al learning

rate: un learning rate pel bucle extern, un learning rate pel bucle intern durant l'entrenament i un learning rate pel bucle intern durant la fase de validació i test. Per defecte, els dos learning rates del bucle intern (el corresponent a la fase d'entrenament i el corresponent a les fases de validació i test) seran iguals, però donen l'opció de poder utilitzar dos valors diferents. Per tal d'analitzar els problemes que això pot comportar, examinarem per separat les conseqüències que té i possibles solucions que podem aplicar tant al bucle intern com al bucle extern. El fet de tenir el learning rate del bucle intern fixat, fa que obliguem al nostre model a aprendre a la mateixa velocitat per cada tasca i per a cada iteració de descens de gradient. Si mantenim aquest hiperparàmetre fixat, estem suposant que totes les tasques aporten la mateixa quantitat d'informació a cada època d'entrenament, però el més intuïtiu és que no totes les tasques siguin igual de significatives. Per altra banda, si deixem el learning rate fixat pel bucle extern (per la meta-optimització) estem fent que el nostre model extregui la mateixa quantitat d'informació de cada època de l'entrenament.

Una bona pràctica és anar reduint el learning rate a mesura que augmenten les èpoques d'entrenament, ja que al principi ens podem permetre el luxe de fer grans passos per a buscar la zona on es troba el mínim, però a mesura que avancem en l'entrenament hauríem de reduir els nostres passos per tal d'assegurar de no saltar-nos el punt on es troba el mínim de la funció objectiu.

5.2 Solucions proposades en la literatura

Un cop presentats els problemes ens preguntem, tenen una solució immediata sense afectar a l'eficàcia del model? Podem proposar una solució sense reduir la capacitat de l'algorisme d'aprendre i generalitzar? Per tal de poder proporcionar solucions i millores d'aquest algorisme analitzarem la literatura que hi ha al respecte per tal d'analitzar cada problema per separat.

El primer problema que hem comentat és que l'algorisme presenta inestabilitats durant l'entrenament si l'arquitectura de la xarxa utilitza strided convolutions. Els autors originals de MAML diuen que l'arquitectura utilitzada per l'entrenament del dataset d'Omniglot utilitza strided convolutions. Tot i així, els autors de [14] van detectar que durant l'entrenament del model no hi havia una progressió de la seva precisió constant, sinó que hi havia algunes èpoques on augmentava i de cop disminuïa.

Els autors de l'article [14] van teoritzar que aquestes inestabilitats eren degudes al fet que els gradients eren passats pels mateixos paràmetres de la xarxa diversos cops, la qual cosa podia acabar provocant el que coneixem com a *exploding gradients* o *vanishing gradients*. Aquests termes fan referència al fet de quan els gradients són multiplicats diversos cops pels paràmetres de la xarxa durant el procés d'optimització, i, per tant, acabem obtenint valors del gradient molt grans (en el cas d'un valor superior a 1, exploding gradients) o gradients molt propers a zero (en el cas de gradients menors que 1, vanishing gradients). Els autors de [14] proposen com a solució d'aquest problema en comptes de calcular el gradient i l'optimització dels paràmetres generals θ utilitzant només el gradient aconseguit al final de cada tasca, utilitzar els gradients de cada tasca

i de cada optimització dins de cada tasca. Per tal de fer això proposen una suma ponderada dels gradients, donant més pes als gradients obtinguts a les últimes optimitzacions de cada tasca. D'una manera més formal, l'optimització dels paràmetres θ quedaria:

$$\theta = \theta - \beta \nabla_\theta \sum_{b=1}^{B} \sum_{i=0}^{N} v_i L_{T_b}(f_{\theta_i^b}) \tag{5}$$

on b denota la tasca sobre la qual estem optimitzant, i denota la iteració en el procés d'optimització en la que ens trobem, v_i és un vector que serà més gran com més gran sigui el valor i, per tant θ_i^b denota els paràmetres de la tasca b després de la i-éssima optimització de descens de gradient.

El segon problema obvi que trobem a l'algorisme de MAML és que comporta un cost computacional elevat, a causa del càlcul de segones derivades que comporta l'algorisme. Els autors de l'article original ja van tenir-ho en compte i van parlar de realitzar una aproximació de primer ordre de MAML, anomenada fo-MAML (*first order MAML*) [4]. Aquesta aproximació consistia en obviar les segones derivades, i treballar amb el mateix algorisme, però sense calcular les segones derivades. Tot i que aquesta primera aproximació redueix el temps d'execució considerablement, també comporta una reducció de la precisió del model, tenint un impacte directe en la capacitat d'adaptació i generalització d'aquest. És per això que els autors de [14] proposen una solució intermèdia. Durant les primeres 50 èpoques d'entrenament proposen utilitzar l'aproximació de primer ordre de MAML de manera que el temps d'execució durant aquestes primeres èpoques serà bastant més reduit, però a la resta utilitzarem l'algorisme estàndard de MAML (calculant les segones derivades). En utilitzar les segones derivades a la segona part de l'entrenament fem que el model no perdi la capacitat de generalitzar que li aporta aquestes segones derivades.

El tercer dels problemes dels que hem parlat és que l'entrenament i les validacions per tal de comprovar el funcionament de l'algorisme és fan sobre tasques que són molt semblants entre elles (com és el cas d'Omniglot) i, per tant, els resultats de la precisió del model acostumen a ser molt alts, la qual cosa implica que no tinguem massa marge per comparar amb altres models. És per això que a l'article [18] proposen utilitzar un dataset format per la unió de diversos benchmarks utilitzats per a classificació d'imatges.

La creació d'aquest nou dataset, anomenat Meta-Dataset, involucra la unió de 10 datasets diferents. Aquests datsets són: Imagenet [16] (ja vist a aquest mateix treball una versió reduïda, Mini-ImageNet), Omniglot, Aircraft [19] (el qual conté imatges corresponents a aeronaus, la gran majoria de les quals són avions), Birds [20] (que conté diverses mostres de diferents tipus d'ocells), Describable Textures [21] (el qual conté diferents imatges corresponents a alguns tipus de textures), Quick Draw [22] (conté imatges de dibuixos fets a mà del joc *Quick Draw!*), Fungi [23] (el qual conté imatges de diferents tipus de fongs), VGG Flower [24] (consisteix en diferents imatges de flors), Traffic Signs [25] (consisteix en diferents senyals de trànsit) i MSCOCO [26] (format per diversos objectes comuns). El fet d'utilitzar tots aquests datasets implica que les classes

sobre les que farem Meta-Learning seran molt diferents entre si i, per tant, podrem valorar millor el flux d'informació i d'experiència prèvia que pot haver-hi entre tasques.

Finalment, per tal de solucionar el problema del learning rate, a l'article [14] proposen dues solucions. La primera d'elles és per solucionar el learning rate intern. Per tal d'evitar que aquest hiperparàmetre sigui compartit igual per totes iteracions de l'optimització, proposen aprendre un learning rate diferent per a cada capa de la xarxa. A més, per a cada learning rate aprés hi haurà N instàncies d'aquest, una per a cada iteració en l'optimització de cada tasca. Per altra banda, per tal de solucionar el problema del learning rate del bucle extern, proposen aplicar un *cosine annealing*. Aquesta tècnica consisteix a variar el learning rate de tal manera que a les primeres èpoques d'entrenament pugui tenir un valor relativament gran i que a mesura que avanci aquest valor pugui anar reduint-se, i més endavant, si és necessari, tornar a augmentar.

A més d'aquestes solucions pels problemes presentats anteriorment, s'han presentat moltes altres millores referents a MAML en la literatura. Els autors de [14] implementen algunes de les millores descrites juntament amb altres que se centren en millorar la velocitat de convergència i l'estabilitat. Al model resultant d'aplicar totes aquestes millores l'anomenen MAML++. A més, a l'article [18], a part de posar a prova l'algorisme amb una nova base de dades, també implementen una millora a l'algorisme de MAML, basant-se en la seva combinació amb les Prototypical Networks. Aquest nou model implementa el mateix algorisme que MAML amb l'única diferència que s'aplica una capa lineal específica per a cada tasca i els seus pesos són inicialitzats a cada episodi amb els paràmetres d'una capa corresponent de les Prototypical Networks. Els gradients flueixen per tal de modificar els paràmetres de les dues xarxes neuronals. Una altra millora que podem trobar és l'anomenada Reptile [27], la qual implementa una aproximació de primer ordre de MAML (fo-MAML) que consisteix en després d'optimitzar els paràmetres en una tasca concreta, moure la inicialització de paràmetres en la mateixa direcció en la qual han evolucionat els paràmetres actualitzats d'aquesta última tasca. Per últim, també podem destacar TSA-MAML [28] que es basa en agrupar les tasques segons la seva similitud i que la inicialització de paràmetres que busca sigui lleugerament diferent per a cada agrupació de tasques. Aquestes són algunes de les variacions i millores proposades per la literatura, tot i que no són les úniques.

6 Conclusions

Un cop analitzat el rendiment de MAML i la base teòrica d'aquest podem concloure que ha estat un algorisme tan important a causa dels bons resultats que aporta enfront de la seva senzillesa. Hem pogut comprovar que el seu rendiment i capacitat de generalitzar és molt superior a les Matching i les Prototypical Networks i, tot i que sigui inferior en quant als resultats obtinguts respecte de les Relation Networks, MAML continua sent un model més senzill computacionalment parlant, que requereix l'estimació de menys paràmetres i

que necessita menys temps d'execució. És per això que l'algorisme que aporta millors resultats enfront de la seva senzillesa és MAML. Això fa que sigui un algorisme relativament fàcil de millorar, tal com hem pogut veure a diversos articles que han implementat versions millorades, com poden ser MAML++ [14], Proto-MAML [18], Reptile [27] o TSA-MAML [28]. A més, quan hem analitzat el seu rendiment en funció del nombre de mostres d'entrenament per classe hem pogut veure que MAML assolia la seva capacitat de màxima generalització en un problema de classificació amb Mini-ImageNet entre les 10 i les 15 mostres per classe, i que pel problema de regressió amb menys de 10 mostres aconseguia predir la natura ondulatòria de la funció sinus, fet que fa que puguem concloure que MAML ha complert el seu objectiu: generalitzar tasques senceres a partir de poques mostres d'entrenament, al contrari que les Matching o les Prototypical Networks, que en el problema de classificaió continuaven extraient informació amb més mostres d'entrenament, però amb un rendiment inferior al de MAML.

Com hem vist hi ha moltes maneres de millorar l'algorisme de MAML i és per això que marca un bon punt de partida per a treballs futurs. En aquest projecte hem vist algunes de les possibles millores que es poden aplicar a MAML, tot i així, el fet de ser un algorisme senzill ha provocat que sorgissin diverses implementacions millorades d'aquest algorisme, tal com hem vist en les seccions anteriors. Aquest treball podria ampliar-se estudiant aquestes millores per tal de veure quines diferències en complexitat i eficàcia presenten entre elles i enfront de l'algorisme original. A més algunes d'aquestes millores no són excloents entre elles, com ja van posar de manifest els autors de [18], que van dir que la implementació del seu Proto-MAML no és excloent a la implementació de MAML++ i, per tant, poden ser combinades entre elles.

Agraïments

Aquest últim any de formació en un grau darrerament creat i en el qual d'aquí unes poques setmanes passaré a formar part de la primera generació de graduats, ha sigut una experiència inoblidable. Vull agrair tot l'esforç fet per l'equip docent dedicat a la creació i consolidació d'aquest grau, el qual m'ha ofert un punt de vista, una formació i una experiència en un món que fa uns pocs anys desconeixia completament. Vull agrair als meus companys i companyes tot l'esforç i el suport durant aquests quatre anys de formació, a més de la meva família, amics, amigues i tothom que ha estat al meu costat durant aquests quatre anys de treball.

Per últim, vull fer una especial menció al Jordi Gonzàlez Sabaté, el meu tutor durant la realització d'aquest treball i sense l'ajuda i els consells del qual res del que he fet aquí hagués estat possible.

Referències

[1] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning, 2016.

[2] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175, 2017.

[3] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[4] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017.

[5] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. *CoRR*, abs/1909.09157, 2019.

[6] F. Hutter, L. Kotthoff, and J. Vanschoren. *Automated Machine Learning: Methods, Systems, Challenges*. The Springer Series on Challenges in Machine Learning. Springer International Publishing, 2019.

[7] Joaquin Vanschoren. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018.

[8] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3), jun 2020.

[9] Etienne Bennequin. Meta-learning algorithms for few-shot computer vision. *CoRR*, abs/1909.13579, 2019.

[10] Sadiq Sani, Nirmalie Wiratunga, Stewart Massie, and Kay Cooper. Matching networks for personalised human activity recognition. CEUR Workshop Proceedings, 2018.

[11] Kuiliang Gao, Bing Liu, Xuchu Yu, Pengqiang Zhang, Xiong Tan, and Yifan Sun. Small sample classification of hyperspectral image using model-agnostic meta-learning algorithm and convolutional neural network. *International Journal of Remote Sensing*, 2021.

[12] Abiola Obamuyide and Andreas Vlachos. Model-agnostic meta-learning for relation classification with limited supervision. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5873–5879, Florence, Italy, July 2019. Association for Computational Linguistics.

[13] Brenden M. Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua B. Tenenbaum. One shot learning of simple visual concepts. In Laura A. Carlson, Christoph Hölscher, and Thomas F. Shipley, editors, *Proceedings of the 33th Annual Meeting of the Cognitive Science Society, CogSci 2011, Boston, Massachusetts, USA, July 20-23, 2011*. cognitivesciencesociety.org, 2011.

[14] Antreas Antoniou, Harrison Edwards, and Amos J. Storkey. How to train your MAML. *CoRR*, abs/1810.09502, 2018.

[15] Pan Zhou, Yingtian Zou, Xiao-Tong Yuan, Jiashi Feng, Caiming Xiong, and Steven Hoi. Task similarity aware meta learning: theory-inspired improvement on MAML. In Cassio de Campos and Marloes H. Maathuis, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 23–33. PMLR, 27–30 Jul 2021.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[17] Sachin Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

[18] Eleni Triantafillou, Tyler Lixuan Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and H. Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *ArXiv*, abs/1903.03096, 2020.

[19] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[20] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge J. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[21] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[22] Takashi Kawashima Jongmin Kim Jonas Jongejan, Henry Rowley and Nick Fox-Gieg. The quick, draw! – a.i. experiment. `quickdraw.withgoogle.com`, 2016.

[23] Brigit Schroeder and Yin Cui. Fgvcx fungi classification challenge 2018. `github.com/visipedia/fgvcx_fungi_comp`, 2018.

[24] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008.

[25] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *International Joint Conference on Neural Networks*, 2013.

[26] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[27] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999, 2018.

[28] Pan Zhou, Yingtian Zou, Xiao-Tong Yuan, Jiashi Feng, Caiming Xiong, and Steven Hoi. Task similarity aware meta learning: theory-inspired improvement on MAML. In Cassio de Campos and Marloes H. Maathuis, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 23–33. PMLR, 27–30 Jul 2021.