Regularisation for Efficient Softmax Parameter Generation in Low-Resource Text Classifiers

Daniel Grießhaber[1] , **Johannes Maucher**[1] , **Ngoc Thang Vu**[2]
[1]Institute for Applied Artificial Intelligence (IAAI)
Stuttgart Media University
Nobelstraße 10, 70569 Stuttgart
[2] Institute for Natural Language Processing (IMS)
University of Stuttgart
Pfaffenwaldring 5B, 70569 Stuttgart
{griesshaber, maucher}@hdm-stuttgart.de, thang.vu@ims.uni-stuttgart.de

Abstract

Meta-learning has made tremendous progress in recent years and was demonstrated to be particularly suitable in low-resource settings where training data is very limited. However, meta-learning models still require large amounts of training tasks to achieve good generalisation. Since labelled training data may be sparse, self-supervision-based approaches are able to further improve performance on downstream tasks. Although no labelled data is necessary for this training, a large corpus of unlabelled text needs to be available. In this paper, we improve on recent advances in meta-learning for natural language models that allow training on a diverse set of training tasks for few-shot, low-resource target tasks. We introduce a way to generate new training data with the need for neither more supervised nor unsupervised datasets. We evaluate the method on a diverse set of NLP tasks and show that the model decreases in performance when trained on this data without further adjustments. Therefore, we introduce and evaluate two methods for regularising the training process and show that they not only improve performance when used in conjunction with the new training data but also improve average performance when training only on the original data, compared to the baseline.

1 Introduction

In recent years, self-supervised methods for training large text models have shown tremendous success in many natural language processing (NLP) tasks by training a general language model without the need for supervised data, followed by a fine-tuning step to train the model for a specific task [Devlin *et al.*, 2019; Radford *et al.*, 2019; Brown *et al.*, 2020]. However, this fine-tuning process still requires large amounts of labelled and task-specific data [Yogatama *et al.*, 2019] which often is not available in real-world scenarios [Tan and Zhang, 2008; Wan, 2008; Salameh *et al.*, 2015; Fang and Cohn, 2017].

Another field that has recently received lots of attention is meta-learning [Schmidhuber, 1987; Bengio *et al.*, 1991; Finn *et al.*, 2017] where the goal is to learn a meta-model on many small tasks so that it can be used to more efficiently train a task-specific target model downstream with only a few labelled training examples. One recent development in this field is an optimisation-based learning method that is able to learn a good initialisation point for an arbitrary model architecture so that the initialised model only needs small adjustments during training to converge and is consequently able to achieve good performance for small training sets [Finn *et al.*, 2017].

Although model agnostic, this technique still requires a fixed model architecture and shares all learned initialisation parameters including the final classification layers. This poses a problem for diverse NLP tasks that can have disjoint label spaces as well as different numbers of possible output classes. To solve this problem, a parameter generator can be used to dynamically generate the initial parameters of the classification layer based on samples of the training data [Bansal *et al.*, 2020a]. This generator is part of the meta-model and is optimised during meta-training to produce good initial parameters that can be fine-tuned using only few training examples from the target task. This adjustment allows the use of all available labelled training data during training, even if the number of classes N does not match the target task. Still, it is beneficial for the average model performance to further increase the amount of training data, especially in low-resource settings [Bansal *et al.*, 2020b].

In this paper, we explore ways to leverage training on additional tasks that are generated from the already available data by changing the training objective. Therefore, these datasets require no additional labelled or unlabelled data. More concretely our contributions are:

- A method to generate tasks from the available data with a new training objective
- We demonstrate that this data can not naively be used for training and therefore
- We introduce two different methods to regularise the training process and

- Demonstrate that these techniques are not only effective while training on the newly generated tasks, but also improve average model performance even when only using the original datasets

We evaluate all methods on a diverse set of few-shot NLP tasks to demonstrate their effectiveness and present a detailed analysis of the effects of the proposed methods.

2 Background

2.1 Meta-Learning

Meta-learning describes the process of training a model to be able to quickly adapt to a new task. This is achieved by learning a meta-task over different tasks which is used to optimise the learning process [Schmidhuber, 1987; Bengio *et al.*, 1991; Xiao *et al.*, 2021] which makes meta-learning particularly suitable for low-resource tasks [Vinyals *et al.*, 2016; Bansal *et al.*, 2020a; Bansal *et al.*, 2021; Wang *et al.*, 2021a; Li and Zhang, 2021; Han *et al.*, 2021]. Often a meta-model is trained to update the parameters of the task-specific models [Hochreiter *et al.*, 2001; Andrychowicz *et al.*, 2016; Li and Malik, 2017] which comes with the disadvantage of introducing new parameters that itself need to be optimised during training and therefore introducing a computational overhead to the overall system.

Recent approaches resolve this weakness by using an optimisation-based learning method [Finn *et al.*, 2017; Antoniou *et al.*, 2019; Bechtle *et al.*, 2021] with the goal to learn a set of model parameters θ_0 that function as an initialisation point for downstream tasks, so that the model can be fine-tuned to a concrete task with only very few training steps. "Model-Agnostic Meta-Learning" (MAML) [Finn *et al.*, 2017] is an example of such an algorithm that does not impose any constraints on the model architecture. In MAML, the initial model parameters θ_0 of a neural network f are learned by sampling a number of training tasks $\mathcal{T}_i$ from a set of tasks $\mathcal{M} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots\}$. For each task, a training (support) set $\mathcal{D}_i^s \sim \mathcal{T}_i$ and a validation (query) set $\mathcal{D}_i^q \sim \mathcal{T}_i$ is sampled. The initial model parameters θ_0 are then updated to minimise a task loss function $\mathcal{L}_i^s$ on $\mathcal{D}_i^s$, yielding adjusted model parameters θ_i^*:

$$\theta_i^* = \theta_0 - \alpha \nabla_{\theta_0} \mathcal{L}_i^s(f_{\theta_0}) \tag{1}$$

where α is the learning rate. This step forms the *inner-loop* or the update of the *fast-parameters*. In the *outer-loop*, the updated parameters θ_i^* are used to calculate an optimised set of initial parameters θ_0 by minimising the meta loss over all sampled tasks:

$$\mathcal{L}_{meta}(\theta_0) = \sum_{b=1}^{B} \mathcal{L}_i^q(f_{\theta_i^*}) \tag{2}$$

$$\theta_0^* = \theta_0 - \beta * \nabla_{\theta_0} \mathcal{L}_{meta}(\theta_0) \tag{3}$$

where $\mathcal{L}_i^q$ is the loss on $\mathcal{D}_i^q$, B is the sample size drawn from $\mathcal{T}_i$ and β the learning rate of the outer-loop. The calculation of the outer-loop gradients requires the calculation of the second-order derivative to propagate the loss through the inner-loop updates which imposes a serious computational cost. Interestingly, Finn *et al.* [2017] empirically show that using a first-order approximation of the meta-gradients still yields comparable model performance.

2.2 Few-Shot Learning for Diverse NLP Tasks

Based on the MAML framework, Bansal *et al.* [2020a] propose a new architecture for few-shot text classification across diverse tasks. While MAML is model-agnostic, it requires a fixed model architecture to tune the initialisation point. In the case of an N-way classifier network trained using categorical cross-entropy however, the last layer of the model needs to have exactly N output neurons. Therefore, the naive MAML optimisation approach limits either the availability of suitable tasks that can be used for meta-training, which is especially problematic for large values of N, or would exclude optimisation of the output layer in the meta-update step for tasks with a different number of classes as the target task. Bansal *et al.* [2020a] therefore introduce a parameter generator g_ψ that predicts a set of initial parameters for the output layer using the first batch of training data from $\mathcal{D}_i^s$ by partitioning the data according to the class labels $C_i^n \leftarrow \{\boldsymbol{x}_j | \boldsymbol{y}_j = n\}$ and then predict a set of weight and bias parameters

$$\boldsymbol{w}_i^n, \boldsymbol{b}_i^n \leftarrow \frac{1}{|C_i^n|} \sum_{\boldsymbol{x}_j \in C_i^n} g_\psi(f_\theta(\boldsymbol{x}_j)) \tag{4}$$

using each partition where f_θ is a pre-trained text-encoder model. The output layers are then formed by stacking the predictions for each partition:

$$\boldsymbol{W}_i \leftarrow [\boldsymbol{w}_i^1; \ldots; \boldsymbol{w}_i^{N_i}] \tag{5}$$

$$\boldsymbol{b}_i \leftarrow [\boldsymbol{b}_i^1; \ldots; \boldsymbol{b}_i^{N_i}] \tag{6}$$

g_ψ is itself a simple multi-layer neural-network whose parameters are optimised in the outer loop step. The authors evaluate this architecture on a diverse set of text classification tasks by using a text classifier architecture that is composed of the text encoder f_θ that feeds into the output layers with the generated parameters. The text encoder is a pre-trained *Bidirectional Encoder Representations from Transformers model* (BERT) [Devlin *et al.*, 2019] that is fine-tuned during meta-training. Bansal *et al.* [2020a] call this architecture *Learning to generate softmax parameters for diverse classification (LEOPARD)*.

Since θ_0 is only updated once for every task $\mathcal{T}_i$, the set of available Tasks $\mathcal{M}$ generally needs to be large, which is a major challenge for meta-learning algorithms [Santoro *et al.*, 2016]. One way to create new tasks is by subsampling a set of labels from tasks with large label spaces [Santoro *et al.*, 2016; Ravi and Larochelle, 2017]. However, while there exist many tasks with a large label space for vision applications [Deng *et al.*, 2009; Krause *et al.*, 2013; Zhou *et al.*, 2017], such datasets are rarer for NLP tasks. Bansal *et al.* [2020b] therefore introduce *Subset Masked Language Modeling Tasks (SMLMT)* to generate tasks from unsupervised data by collecting sentences containing N randomly chosen words from the vocabulary and masking all occurrences of those words in k sampled sentences containing one of the N words to create an arbitrary k-shot N-way classification task. The authors show that this approach yields an average improvement in classification accuracy of 3% compared to LEOPARD when this self-supervised training data is used jointly with available supervised data.

3 Methods

3.1 Generation of Training Tasks

We propose yet another method of generating new training tasks that does not rely on new, unsupervised data by using the task affiliation of a sample as its label. This means a new N-way task can be created by sampling N tasks $\mathcal{T}_{n1}, \dots, \mathcal{T}_{nN}$ from $\mathcal{M}$. Each datum $\boldsymbol{x}_{n_i}$ from each task is then assigned the label $\boldsymbol{y}_{n_i} = n$, so that the new task $\mathcal{T}_c$ becomes a problem of N-way dataset association. We denote the set of possible tasks that can be generated by using all N-combinations of the datasets used in $\mathcal{M}$ by $\mathcal{M}_{cN}$ while the original tasks are denoted by $\mathcal{M}_o$.

3.2 Regularise Parameter Generation for New Datasets

As described in section 2.2, parameter generation uses a single batch of training data. With the introduction of new tasks from $\mathcal{M}_{cN}$ during training, there is the potential for a sub-optimal sampling of the first batch of training data in $\mathcal{T}_i$ used as the input to the parameter generator g_ψ where most of the samples in C_i^n stem from a single class m in the task j from $\mathcal{M}_o$ that is part of the combination to create $\mathcal{T}_j$ and therefore $C_i^n \approx C_j^m$. Assuming a fully deterministic f_θ and g_ψ, this would generate the same parameters for classes m and n, yielding a classification layer that is in the extreme case equal to the classification layer for $\mathcal{T}_j$. Therefore, we propose 2 ways of regularising the generation and training of the classification parameters.

Attention for Weighted Average Inputs

We replace the simple mean aggregation on the output of g_ψ in equation 4 with a weighted average. We calculate the weights using attention [Bahdanau *et al.*, 2015; Cheng *et al.*, 2016], specifically a fixed-query attention [Sukhbaatar *et al.*, 2015; Yang *et al.*, 2016] to compute a weight for each sample. Attention was successfully used to focus on a partial context in long sequences in NLP tasks [Luong *et al.*, 2015; Yang *et al.*, 2016; Vaswani *et al.*, 2017]. Our motivation is for the network to weigh outputs from the generator more if they represent a better initial classification parameter for the current task and therefore counteract the sub-optimal selection of samples. Specifically, we replace equation 4 with an attention mechanism [Bahdanau *et al.*, 2015]:

$$\boldsymbol{u}_j = \tanh(\boldsymbol{W}_a\, f_\theta(\boldsymbol{x}_j) + \boldsymbol{b}_a) \tag{7}$$

$$\boldsymbol{\alpha}_j = \frac{e^{\boldsymbol{u}_j^\intercal \boldsymbol{u}_q}}{\sum_j e^{\boldsymbol{u}_j^\intercal \boldsymbol{u}_q}} \tag{8}$$

$$\boldsymbol{w}_i^n, \boldsymbol{b}_i^n \leftarrow \sum_{j=0}^{|C_i^n|} \boldsymbol{\alpha}_j * g_\psi(f_\theta(\boldsymbol{x_j})) \tag{9}$$

First, the output of the text-encoder f_θ is embedded into a hidden representation $\boldsymbol{u}_j$ through a single layer neural network with $\tanh$ activation (7). To calculate the attention score of an input sample $\boldsymbol{\alpha}_j$ this hidden representation is then multiplied with a fixed query-vector $\boldsymbol{u}_q$ and normalised using softmax to ensure $\sum_j \boldsymbol{\alpha}_j = 1$ (8). The query-vector $\boldsymbol{u}_q$ is tuned jointly with $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ during meta-training. Finally, the calculated attention scores are used to weigh the output of the parameter generator g_ψ for each sample (9).

Triplet-Loss

During classification, the generated parameters of the output layer form a multi-dimensional classification boundary. Since in a low-resource setting the initial weights of the output layers W_i, b_i do not have much training data to adapt to, not only is a good initialisation of the parameters essential, but also a quick adaption of those parameters to maximise the classification boundary. While the original LEOPARD architecture uses cross-entropy loss ($\mathcal{L}_{ce}$) to calculate the gradients, we propose the use of the triplet loss function [Schultz and Joachims, 2003; Chechik *et al.*, 2010; Schroff *et al.*, 2015] denoted by $\mathcal{L}_t$ to calculate gradients that help to define an effective classification boundary more quickly by minimising the distance between an input and a sample of the same class (positive) while maximising the distance to samples from different classes (negative). The selection of positive and negative samples is called triplet mining. We use semi-hard triplet mining on mini-batches with a margin of 1.0. If no negative sample can be found that satisfies the semi-hard negative constraint of having a distance larger than the positive distance plus margin, the largest negative distance is used.

Since the triplet loss requires the mining of *good* positive and negative samples which may not always be available in each mini-batch, we use a mixed approach ($\mathcal{L}_{m\lambda}$) that combines triplet loss with cross-entropy:

$$\mathcal{L}_{m\lambda} = \mathcal{L}_t * \lambda + \mathcal{L}_{ce} * (1 - \lambda) \tag{10}$$

where λ is a hyper-parameter of the model that needs tuning and can be used to weigh the influence of each loss function for the overall loss.

Note that both approaches, while introduced to counteract the problems described above, are used during the training of tasks from $\mathcal{M}_o$ and $\mathcal{M}_{cN}$.

4 Experimental Setup

4.1 Training Tasks

We follow Bansal *et al.* [2020a] in the selection of a subset of datasets from the GLUE [Wang *et al.*, 2018] meta dataset, specifically the MNLI (matched and mismatched), MRPC, QNLI, QQP, RTE, SST-2 and SNLI datasets [Bowman *et al.*, 2015] for training the meta-model. STS-B is excluded due to it being a regression task and WNLI for its small size. For tasks with $N>2$ classes, a separate train set for every 2-combination of classes is generated to increase the number of tasks that can be used during meta-training.

4.2 Evaluation Tasks

Evaluation for each experiment is performed on a diverse set of tasks with disjoint label spaces and different numbers of classes.

From the Amazon Review Corpus [Blitzer *et al.*, 2007], we use the Product Categories `Books`, `DVD`, `Electronics` and `Kitchen`. Each dataset contains reviews for products of the category together with an associated product rating. While the original data is separated into 5 rating classes, we map those into N=3 distinct labels $\{\leq 2, =4, =5\}$.

The *CoNLL*-2003 shared task [Tjong Kim Sang and De Meulder, 2003] is a named entity recognition task. Each input consists of a sentence and a named entity that appears in this sentence with the label being one of N=4 different entity types.

Bansal *et al.* [2020a] describe a set of tasks that were sourced from crowdflower which are not available from the provided source anymore after the acquisition of the company. Therefore, we provide new sources. All datasets consist of tweets in English with different label spaces depending on the task. The *Airline* dataset[1] consists of tweets about north American airlines with N=3 labels to classify the sentiment of the text. The *Disaster* dataset[2] contains tweets with keywords such as `crash` or `bush fire` with N=2 classes indicating whether or not the input is about a real disaster. The *Emotion* dataset[3] contains N=13 different emotions as a classification target. While other authors previously have reduced this label space by combining and selecting labels due to strong similarities between some classes [Bouazizi and Ohtsuki, 2016], we follow Bansal *et al.* [2020a] and use the unaltered selection. The *Political Audience*, *Political Bias* and *Political Message*[4] tasks all use the same input texts but have different class labels depending on the intended audience (N=2, {`national`, `constituency`}), the bias (N=2, {`neutral`, `partisan`}) or message (N=9, {`policy`, `personal`, `...`}) of the input.

A k-shot subset of a dataset is created by choosing k random samples from each of the N classes. We aggregate the mean accuracy for 10 different subsets for each dataset with $k \in \{4, 8, 16\}$ and report the average accuracy and the standard deviation between runs.

4.3 Baseline Experiments

Although Bansal *et al.* [2020a] provide the sub-sampling of test datasets used in their experiments online, due to corruptions in some of the files we recreated the task data, resulting in a different selection of data points.

To be able to compare any improvements of our approaches, we use Bansal *et al.* [2020a] as a baseline and evaluate on the same datasets. We copy the previously published results in column *LEOPARD*. However due to the large variance in the reported accuracies and to preclude any possible differences in the selection of samples for the few-shot subsets we also provide the results of the unmodified architecture on our datasets in the *Baseline* column.

[1]https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

[2]https://www.kaggle.com/datasets/vstepanenko/disaster-tweets

[3]https://www.kaggle.com/datasets/pashupatigupta/emotion-detection-from-text

[4]https://www.kaggle.com/datasets/crowdflower/political-social-media-posts

	Dataset		Loss		α
	$\mathcal{M}_o$	$\mathcal{M}_{c2}$	$\mathcal{L}_{ce}$	$\mathcal{L}_{m\lambda}$	
Leopard	●		●		
Baseline	●		●		
$\boldsymbol{A}$	●			0.5	
$\boldsymbol{B}$	●		●		●
$\boldsymbol{\Gamma}$	●			0.5	●
$\boldsymbol{\Delta}$	●	●	●		
$\boldsymbol{E}$	●	●	●		●
$\boldsymbol{Z}$	●	●		0.5	
$\boldsymbol{H}$	●	●		0.5	●
$\boldsymbol{\Theta}$	●			0.1	
$\boldsymbol{I}$	●			1.0	

Table 1: Experiment Setup Configurations

4.4 Experiment Configurations

Table 1 shows a matrix of model and dataset configurations used in the experiments. The *Dataset* column describes which datasets were used while training the model showing whether additionally generated training data was available or not. The *Loss* column indicates whether cross-entropy ($\mathcal{L}_{ce}$) or the mixed-loss approach described in section 3.2 ($\mathcal{L}_{m\lambda}$) is used where the number is the set value for parameter λ in the experiment. A dot in column α indicates that attention was used in the parameter generator to calculate sample weights as described in section 3.2.

In experiment $\boldsymbol{A}$, $\boldsymbol{\Theta}$ and $\boldsymbol{I}$ the triplet-loss function with different values for $\lambda = \{0.1, 0.5, 1.0\}$ and in experiment $\boldsymbol{B}$ the attention weighting is used during training on the original set of tasks respectively to examine the effect of the proposed regularisation methods without introducing new tasks. In experiment $\boldsymbol{\Delta}$ new tasks are introduced, however only the cross-entropy term is used in the loss function to test whether the generated data can naively be used to further train the meta-model on more tasks. Experiment $\boldsymbol{Z}$ uses the additional training tasks in conjunction with the mixed loss function and $\lambda = 0.5$ to determine the effectiveness of the triplet-loss regularisation when combining the task sets. Experiment $\boldsymbol{H}$ combines attention-based weighting with mixed-loss to determine if further improvements can be achieved by combining the approaches.

The implementation used for all experiments in this work is available for reference online[5].

5 Results

This section presents the results of the performed experiments in an objective manner. A thorough analysis of the behaviour of the proposed techniques is presented in the next section.

Table 2 shows the accuracy of the different model configurations presented in the previous section averaged over all datasets.

[5]https://gitlab.mi.hdm-stuttgart.de/griesshaber/metanlp

[6]Since we calculate the average over all datasets while Bansal *et al.* [2020a] reserve some datasets for domain transfer experiments, our reported average differs from the reported results, although the accuracy of the separate experiments is approximately the same.

k	LEOPARD[6]	Baseline[6]	$\boldsymbol{A}$	$\boldsymbol{B}$	$\boldsymbol{\Gamma}$	$\boldsymbol{\Delta}$	$\boldsymbol{E}$	$\boldsymbol{Z}$	$\boldsymbol{H}$	$\boldsymbol{\Theta}$	$\boldsymbol{I}$
4	46.15	45.97	47.64	47.58	47.50	43.64	48.13	**48.59**	48.51	46.04	37.73
8	50.24	50.73	51.58	51.37	51.46	48.62	52.67	**53.47**	53.26	51.28	40.88
16	53.12	52.06	53.21	53.60	52.85	52.63	54.30	**55.30**	54.69	52.79	41.39

Table 2: Evaluation results of the different approaches. For brevity, only the average results over all tasks are presented, while a listing of individual task results is presented in the appendix. The configuration of each experiment column is depicted in table 1

5.1 Baseline Results

The aggregated results of our Baseline match the reported results by Bansal *et al.* [2020a] closely. Therefore, we assume any variation in data selection is unimportant when evaluating the averaged model performance. However, we point out that some individual experiments diverge between the two setups, although in most cases the deviation is within 1 standard deviation from the mean.

5.2 Effect of Regularisation on Baseline Datasets

Experiment $\boldsymbol{A}$ shows an overall improvement to the baseline, indicating that the mixed triplet-loss strategy already helps in the low-resource classification case without introducing new tasks. A similar behaviour, although less pronounced in many experiments, can be observed in experiment $\boldsymbol{B}$ indicating the attention-based sample weighting to have the hypothesised effect of helping the parameter generator to weight the samples used in contrast to using an unweighted average. Combining both methods in experiment $\boldsymbol{\Gamma}$, a slight decrease in performance compared to $\boldsymbol{A}$ and $\boldsymbol{B}$ can be observed. This result indicates that the selection of training samples used to generate parameters is essential for a good initialisation of the classification head of the model.

5.3 Training on Generated Datasets

In experiment $\boldsymbol{\Delta}$ the decrease in performance is an indication that, without additional regularisation, the introduced data hinders the meta-learning process. We use a qualitative and quantitative exploration in the next subsection to argue why this behaviour emerges. Experiment $\boldsymbol{E}$ and $\boldsymbol{Z}$ show the results of training on the same data but using the attention and the mixed triplet loss instead of only cross-entropy respectively, which shows a clear improvement, not only with respect to experiment $\boldsymbol{\Delta}$ but even to both baselines. This is an indication that the generated training tasks based on the available data can be used to improve meta-training and thus help the model to initialise with better initial parameters and adapt more quickly. In experiment $\boldsymbol{H}$ both regularisation techniques are combined, however, performance across all configurations of k decreases slightly compared to experiment $\boldsymbol{Z}$. We conclude that both techniques are effective because they regularise the training process, however applying both methods does not further improve, and in fact hinders the performance due to the added complexity. This is also supported by the performance decrease in experiment $\boldsymbol{\Gamma}$ described earlier.

5.4 Pure- and Mixed Triplet Loss Approaches

Columns $\boldsymbol{A}$, $\boldsymbol{\Theta}$ and $\boldsymbol{I}$ show the importance of the selection of λ in the mixed training loss $L_{m\lambda}$. With a value of $\lambda = 0.1$ in experiment $\boldsymbol{\Theta}$, the results are close to the baseline with a maximum change of <1%. Experiment $\boldsymbol{I}$ with $\lambda = 1.0$ shows a stark decrease in performance. In the individual experiments one can observe that the accuracy approaches random guessing ($\approx \frac{1}{N}$). Only in experiment $\boldsymbol{A}$ with $\lambda = 0.5$ an improvement in task accuracy can be observed. The results are discussed further in the next section.

5.5 Comparison

The best average accuracy across all experiments for all tested settings of k is attained in experiment $\boldsymbol{Z}$ using the additional data with $\mathcal{L}_{m0.5}$ as loss function. This setting is able to achieve an average improvement of +2.87% (+2.62%, +2.74%, +3.24%) over our Baseline results and +2.62% (+2.44%, +3.23%, +2.18%) over the results reported in [Bansal *et al.*, 2020a]. Without additional data, using $\mathcal{L}_{m0.5}$ (A) we attain an improvement of +1.22% whereas the attention weighted parameter generation (B) achieves +1.28%.

6 Evaluation & Analysis

This section quantitatively and qualitatively explores the effects of the proposed techniques for regularisation on the trained model to better understand their effectiveness and importance.

6.1 Effect of λ for Mixed-Loss Experiments

Experiments $\boldsymbol{A}$, $\boldsymbol{\Theta}$ and $\boldsymbol{I}$ in table 2 show the average performance of the meta-models with different configurations for the hyper-parameter λ (0.5, 0.1 and 1.0 respectively).

These results give insight into the importance of tuning the λ parameter. With a value of $\lambda = 0.1$ the effect of the triplet loss seems negligible as the resulting average accuracy is close to the performance of the baseline model for all configurations of k. Using a pure triplet loss function ($\lambda = 1.0$) during training however does result in the performance degrading close to random guessing, indicating that the meta-model fails to learn a good initialisation for θ_0 and also does not allow efficient optimisation during fine-tuning. This shows using only $\mathcal{L}_t$ fails to provide good feedback for optimising the model. We conjecture this is most likely due to the need to mine good triplets, which may be challenging for some low-resource tasks.

The results for $\lambda = 0.5$ indicate that this value seems to provide a good balance between optimising the learned representations for the triplet objective of moving anchors close to positive samples but far from negative samples while not completely relying on the mined triplets which seem to be insufficient as indicated by the pure triplet loss experiment.

6.2 Quantitative Analysis of the Regularisation Approaches

As stated in section 3.2, the motivation for the evaluated attention mechanism and triplet loss regularisation techniques is to help with sub-optimal random sampling during parameter generation. Therefore, we analyse the effectiveness of regularisation on a learned few-shot classifier which is trained to predict dataset association between the *Rating Books* and *Rating Kitchen* datasets ($N = 2$) with $k = 16$. Both datasets are not part of $\mathcal{M}_o$ and $\mathcal{M}_{c2}$ and are tasks of sentiment classification, thus the meta-model is trained on other tasks with a similar objective. Since the objective of the test task however is domain association, the fine-tuned classifier should have good separation between the samples of the two datasets at the output (exa-class), whereas artefacts of the sentiment task association (inter-class) may indicate a sub-optimal initialisation of the model on the sentiment classification task.

To quantitatively measure the inter- and exa-class separation, we perform K-means clustering on the embedded model output and calculate the adjusted mutual information (AMI) between the clustering and the true labels. The exa-class AMI (AMI_e) can be calculated directly, whereas, for the inter-class AMI (AMI_i), we perform a separate clustering for each class and report the average value.

Table 3 shows the results for 3 different loss function configurations. The results for the cross-entropy loss ($\mathcal{L}_{ce}$) show a higher adjusted mutual information than any of the triplet loss experiments for the inter-class clustering, indicating that the output representation contains artefacts of the sentiment task. The results of the mixed-loss experiments show a decrease in inter-class AMI with higher values for λ, supporting the hypothesis that the mixed-loss objective regularises the training by providing better feedback for the current task compared to a pure cross-entropy measure. However, note that for $\lambda = 1$ the exa-class AMI is lowest, indicating a bad separation between classes in the output representations. This is also supported by the results in section 6.1 that showed that a pure triplet loss function is unable to train a few-shot classifier.

The results for the experiment using the attention-based sample weighting ($\mathcal{L}_{ce} + \alpha$) show that this method also successfully improves exa-class separation, however, inter-class separation is much less effectively reduced. This is expected as the triplet-loss actively tries to minimise the distance to positive samples and maximise the distance to negative samples, while the motivation for attention-based weighting is only to weigh samples more or less depending on their relevance to the task.

6.3 Qualitative Evaluation of the Embedding and Output Spaces

Figure 1 shows a visualisation of the classification boundary of the evaluated classifier. The visualisation is created by embedding the high-dimensional output of the text encoder into a 2-dimensional space using t-SNE [van der Maaten and Hinton, 2008] to get a vector $[\boldsymbol{x}_1 \quad \boldsymbol{x}_2]$ for each data point. This vector is then extended by another component that represents the 1-d normalised softmax output of the trained network $\boldsymbol{x}_3 = ((\boldsymbol{\chi}_0 - \boldsymbol{\chi}_1) + 1)/2$ where χ_n is the n-th component of the softmax-activated model output to get a value in $[0, 1]$. The resulting 3-dimensional vector is then coloured red or blue depending on the source dataset while colour saturation shows different classes within the dataset. Therefore, a higher euclidean distance between red and blue clusters indicates a higher exa-class separation while intra-class separation is represented as the distance between clusters of differently shaded dots in the same colour.

	AMI_i	AMI_e
$\mathcal{L}_{ce}$	0.153	0.350
$\mathcal{L}_{m0.1}$	0.017	0.439
$\mathcal{L}_{m0.5}$	0.015	0.688
$\mathcal{L}_{m1.0}$	0.011	0.323
$\mathcal{L}_{ce} + \alpha$	0.045	0.455

Table 3: Adjusted Mutual Information of k-Means Clustering performed on the Embeddings with different Loss Function Configurations.

Subfigure 1b shows a lower variance along the 3rd axis with samples close around the value 0.5, indicating a smaller learned exa-class boundary. Furthermore, within each class, clusters of the same saturation can be seen. This may indicate that the encoder after training still produces embeddings that differ between inter-dataset classes (low intra-class separation).

In contrast, subfigure 1a shows a much broader separation between classes with almost no data points near the classification border. The visualisation also shows less clustering of data points with similar saturation, indicating that this model produces good embeddings for the task without showing artefacts of different classification tasks, supporting the quantitative results.

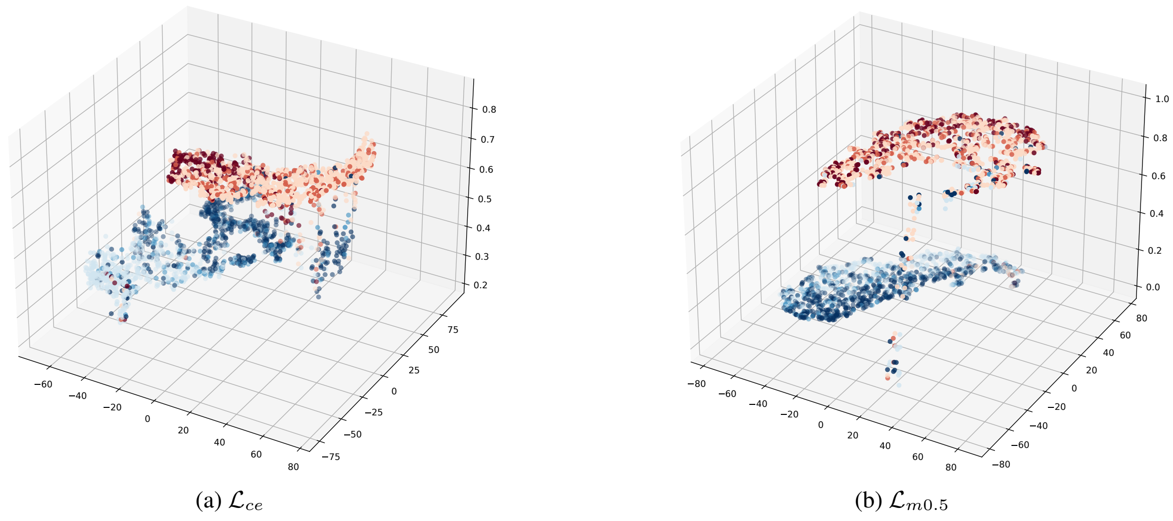
We interpret this visualisation as another indicator that shows the importance of regularisation when using the generated datasets.

7 Related Work

7.1 Meta-Learning

Methods for meta-learning can generally be classified into 3 different approaches; *metric-based* where the model learns an embedding of data points that is trained by reducing a distance metric between related data points [Vinyals *et al.*, 2016; Snell *et al.*, 2017; Sung *et al.*, 2018], *model-based* where a meta-model predicts parameters that are used to initialise a task-specific model [Santoro *et al.*, 2016; Munkhdalai and Yu, 2017] and *optimisation-based* where task specific parameters are obtained through optimisation [Ravi and Larochelle, 2017; Finn *et al.*, 2017; Nichol *et al.*, 2018].

Building on the MAML framework presented in Finn *et al.* [2017], Bansal *et al.* [2020a] introduce the LEOPARD architecture by adding a model-based parameter generator for the classification layer to the otherwise optimisation-based MAML architecture.



(a) $\mathcal{L}_{ce}$ (b) $\mathcal{L}_{m0.5}$

Figure 1: Visualisation of the class boundary on a class association task without (a) and with (b) regularisation using the mixed triplet loss strategy.

7.2 Using Additional Data for Training

In Bansal *et al.* [2020a] they show that an increase of available training data is beneficial for this architecture and create this data using a self-supervised approach called SMLMT. Bansal *et al.* [2021] further improves on those results by tuning the distributions of the sampled self-supervised tasks. In contrast, our approach does not require any additional data for further training. Li and Zhang [2021] use the method presented in Bansal *et al.* [2020a] for generating tasks with a metric-based prototypical network [Snell *et al.*, 2017]. They show that the additional self-supervised training data is able to improve downstream performance when used for meta-learning as well.

7.3 Regularisation of Meta-Learned NLP Classifiers

Bao *et al.* [2020] use an attention mechanism based on word statistics that is trained on meta tasks to aggregate pre-trained embeddings for direct classification using a ridge regressor [Bertinetto *et al.*, 2019] instead of predicting softmax parameters. This method requires the meta-tasks to have related, although disjoint classes from the target data. Han *et al.* [2021] therefore introduce a domain-adversarial approach [Ganin *et al.*, 2016] to generate domain-invariant features in the sentence embeddings.

Liu *et al.* [2020] empirically show that MAML for NLP applications works best when the general language model is not solely trained during meta-training and the meta-tasks are dissimilar to one another. Wang *et al.* [2021b] introduce a variance reduction based on the recursive stochastic momentum technique [Cutkosky and Orabona, 2019] to regularise the meta-gradients during training to prevent overfitting on specific tasks.

8 Limitations

The performed experiments were using English datasets exclusively. This choice was made to ensure some comparability between the baseline results presented by Bansal *et al.* [2020a], the availability of the datasets and the authors' familiarity with the language. However, there exists prior work that shows the effectiveness of both MAML [Zhang *et al.*, 2020] and BERT [Pires *et al.*, 2019; Cui *et al.*, 2021] for different languages.

The process of training a new meta-learning model is relatively expensive since the architecture is quite deep with 110 million parameters only in the BERT text encoder. To train those parameters, a high number of outer-loop training steps need to be performed, each requiring multiple inner-loop optimisation steps. Due to this cost, an extensive hyper-parameter search running many configurations may be prohibitively expensive. Instead, hyper-parameters can be learned as part of the meta-learning procedure as presented in Bansal *et al.* [2020a]. We performed our experiments on compute nodes with 4x NVIDIA 2080 Ti GPUs where training took 72 hours per experiment with the original dataset and an additional 96 hours for the combined dataset.

9 Conclusion

In this paper, we presented a way to generate new training tasks that only depend on the training data already present and can be used to further train a meta NLP classifier. We also showed that without adjustments, these tasks can hinder the average model performance in low-resource settings and therefore require further adaption of the training procedure to circumvent the negative effects. We propose two methods that can be used to regularise the training and prove their effectiveness on both, training with and without the generated tasks to improve average model performance on a wide variety of low-resource evaluation tasks. We then examined the effect of the proposed methods on the trained model using qualitative and quantitative evaluations as well as the effect of the introduced hyper-parameters.

References

[Andrychowicz *et al.*, 2016] Marcin Andrychowicz, Misha Denil, Sergio Gómez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *NIPS*, 2016.

[Antoniou *et al.*, 2019] Antreas Antoniou, Harrison Edwards, and Amos J. Storkey. How to train your MAML. In *ICLR*, 2019.

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[Bansal *et al.*, 2020a] Trapit Bansal, Rishikesh Jha, and Andrew McCallum. Learning to few-shot learn across diverse natural language classification tasks. In *ICCL*, 2020.

[Bansal *et al.*, 2020b] Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. Self-supervised meta-learning for few-shot natural language classification tasks. In *EMNLP*, 2020.

[Bansal *et al.*, 2021] Trapit Bansal, Karthick Prasad Gunasekaran, Tong Wang, Tsendsuren Munkhdalai, and Andrew McCallum. Diverse distributions of self-supervised tasks for meta-learning in NLP. In *EMNLP*, 2021.

[Bao *et al.*, 2020] Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. Few-shot text classification with distributional signatures. In *International Conference on Learning Representations*, 2020.

[Bechtle *et al.*, 2021] Sarah Bechtle, Artem Molchanov, Yevgen Chebotar, Edward Grefenstette, Ludovic Righetti, Gaurav Sukhatme, and Franziska Meier. Meta-learning via learned loss. In *ICPR*, 2021.

[Bengio *et al.*, 1991] Y. Bengio, S. Bengio, and J. Cloutier. Learning a synaptic learning rule. In *IJCNN*, 1991.

[Bertinetto *et al.*, 2019] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019.

[Blitzer *et al.*, 2007] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[Bouazizi and Ohtsuki, 2016] Mondher Bouazizi and Tomoaki Ohtsuki. Sentiment analysis: From binary to multiclass classification: A pattern-based approach for multiclass sentiment analysis in twitter. In *IEEE ICC*, 2016.

[Bowman *et al.*, 2015] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *ACL*, 2015.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NIPS*, 2020.

[Chechik *et al.*, 2010] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *JMLR*, 2010.

[Cheng *et al.*, 2016] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. In *EMNLP*, 2016.

[Cui *et al.*, 2021] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. Pre-training with whole word masking for chinese BERT. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 2021.

[Cutkosky and Orabona, 2019] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex SGD. In *NIPS*, 2019.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*, 2009.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[Fang and Cohn, 2017] Meng Fang and Trevor Cohn. Model transfer for tagging low-resource languages using a bilingual dictionary. In *ACL*, 2017.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *PMLR*, 2017.

[Ganin *et al.*, 2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[Han *et al.*, 2021] Chengcheng Han, Zeqiu Fan, Dongxiang Zhang, Minghui Qiu, Ming Gao, and Aoying Zhou. Meta-learning adversarial domain adaptation network for few-shot text classification. In *IJCNLP*, 2021.

[Hochreiter *et al.*, 2001] Sepp Hochreiter, A. Steven Younger, and Peter R. Conwell. Learning to learn using gradient descent. In *ICANN*, 2001.

[Krause *et al.*, 2013] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE Workshop on 3D Representation and Recognition, 3dRR-13 2013*, 2013.

[Li and Malik, 2017] Ke Li and Jitendra Malik. Learning to optimize. In *ICLR*, 2017.

[Li and Zhang, 2021] Yue Li and Jiong Zhang. Semi-supervised meta-learning for cross-domain few-shot intent classification. In *1st Workshop on Meta Learning and Its Applications to Natural Language Processing*, 2021.

[Liu *et al.*, 2020] Zequn Liu, Ruiyi Zhang, Yiping Song, and Ming Zhang. When does MAML work the best? an empirical study on model-agnostic meta-learning in NLP applications. *CoRR*, abs/2005.11700, 2020.

[Luong *et al.*, 2015] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, 2015.

[Munkhdalai and Yu, 2017] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *PMLR*, 2017.

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

[Pires *et al.*, 2019] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *ACL*, 2019.

[Radford *et al.*, 2019] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

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

[Salameh *et al.*, 2015] Mohammad Salameh, Saif Mohammad, and Svetlana Kiritchenko. Sentiment after translation: A case-study on arabic social media posts. In *NAACL*, 2015.

[Santoro *et al.*, 2016] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, 2016.

[Schmidhuber, 1987] Jurgen Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 1987.

[Schroff *et al.*, 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. *IEEE CVPR*, 2015.

[Schultz and Joachims, 2003] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *NIPS*, 2003.

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017.

[Sukhbaatar *et al.*, 2015] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NIPS*, 2015.

[Sung *et al.*, 2018] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip Torr, and Timothy Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE CVPR*, 2018.

[Tan and Zhang, 2008] Songbo Tan and Jin Zhang. An empirical study of sentiment analysis for chinese documents. *Expert Systems with Applications*, 2008.

[Tjong Kim Sang and De Meulder, 2003] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *HLT-NAACL*, 2003.

[van der Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Viualizing data using t-SNE. *JMLR*, 2008.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[Vinyals *et al.*, 2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NIPS*, 2016.

[Wan, 2008] Xiaojun Wan. Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *EMNLP*, 2008.

[Wang *et al.*, 2018] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *EMNLP Workshop BlackboxNLP*, 2018.

[Wang *et al.*, 2021a] Haoxiang Wang, Han Zhao, and Bo Li. Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation. In *ICML*, 2021.

[Wang *et al.*, 2021b] Lingxiao Wang, Kevin Huang, Tengyu Ma, Quanquan Gu, and Jing Huang. Variance-reduced first-order meta-learning for natural language processing tasks. In *NAACL*, 2021.

[Xiao *et al.*, 2021] Yubei Xiao, Ke Gong, Pan Zhou, Guolin Zheng, Xiaodan Liang, and Liang Lin. Adversarial meta sampling for multilingual low-resource speech recognition. *AAAI*, 2021.

[Yang *et al.*, 2016] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *NAACL*, 2016.

[Yogatama *et al.*, 2019] Dani Yogatama, Cyprien de Masson d'Autume, Jerome T. Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. Learning and evaluating general linguistic intelligence. *CoRR*, 2019.

[Zhang *et al.*, 2020] Xuejun Zhang, Yujiang Li, Pengyuan Zhang, and Yonghong Yan. Lingual-agnostic meta-learning for low-resource part-of-speech tagging. In *ACL ICITS: IoT and Smart City*, 2020.

[Zhou *et al.*, 2017] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.