PROTAUGMENT: Unsupervised diverse short-texts paraphrasing for intent detection meta-learning

Thomas Dopierre[12] and **Christophe Gravier**[1] and
and **Wilfried Logerais**[2]

[1]Laboratoire Hubert Curien
UMR CNRS 5516
Université Jean Monnet
Saint-Étienne, France
firstname.lastname@univ-st-etienne.fr

[2]Meetic
Paris, France
{t.dopierre,w.logerais}@meetic-corp.com

Abstract

Recent research considers few-shot intent detection as a meta-learning problem: the model is *learning to learn* from a consecutive set of small tasks named episodes. In this work, we propose PROTAUGMENT, a meta-learning algorithm for short texts classification applied to the intent detection task. PROTAUGMENT is a novel extension of Prototypical Networks (Snell et al., 2017) that limits over-fitting on the bias introduced by the few-shots classification objective at each episode. It relies on diverse paraphrasing: a conditional language model is first fine-tuned for paraphrasing, and diversity is later introduced at the decoding stage at each meta-learning episode. The diverse paraphrasing is unsupervised as it is applied to unlabelled data and then fueled to the Prototypical Network training objective as a consistency loss. PROTAUGMENT is the state-of-the-art method for intent detection meta-learning, at no extra labeling efforts and without the need to fine-tune a conditional language model on a given application domain.

1 Introduction

Intent detection, a sub-field of text classification, involves classifying user-generated short-texts into intent classes, usually for conversational agents applications (Casanueva et al., 2020). Since conversational agent applications are domain-specific, intent detection is a challenging task because of labeled data scarcity and the number of classes (intents) it usually involves (Dopierre et al., 2020). As a consequence, recent research (Snell et al., 2017; Ren et al., 2018) considers few-shot intent detection as a meta-learning problem: the model is trained to classify user utterances from a consecutive set of small tasks named episodes. Each episode contains a limited number of C classes alongside a limited number of K labeled data for each of the C classes – this is usually referred to as a C-way K-shots setup. At test time, the algorithm is evaluated on classes that were not seen during training. That is the reason why meta-learning is sometimes referred to as *learning to learn*: it mimics human abilities to learn iteratively from different and small tasks. Meta-learning has successfully been applied to a wide set of NLP tasks: hypernym detection (Yu et al., 2020), low resource machine translation (Gu et al., 2018), machine understanding tasks (Dou et al., 2019) or structured query generation (Huang et al., 2018). Most meta-learning algorithms (Section 2) were developed in the course of the last 5 years. It has recently been empirically demonstrated that comparative studies in follow-up papers of (Snell et al., 2017) are debatable – for short texts classification – because of the two following main issues (Dopierre et al., 2021). First, comparative studies involve simple and limited datasets in terms of number and separability of classes (SNIPS (Coucke et al., 2018), a very popular dataset, includes only 7 classes, with the current best model performing over 99% accuracy (Cao et al., 2020)). Second, as we further better understand (Niven and Kao, 2019), fine-tune (Liu et al., 2019b; Hao et al., 2020) and refine (Khetan and Karnin, 2020) BERT-derived models, it is not clear if the different meta-learning frameworks can be considered state-of-the-art due to their architecture or due to the improvements of available text encoders at the time of conception. (Dopierre et al., 2021) concludes that Prototypical Networks (Snell et al., 2017) (that were using LSTM-based text encoders when introduced in NLP) are actually the state-of-the-art for intent detection when equipped with a fine-tuned BERT text encoder model. Ultimately, improving Prototypical Networks have therefore been proven to be a very challenging task in reality.

A cornerstone challenge is that meta-learning models can easily overfit on the biased distribution

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2454–2466
August 1–6, 2021.

introduced by a few training examples (Yang et al., 2021). In order to prevent overfitting and inspired by (Xie et al., 2020), we introduce an unsupervised diverse paraphrasing loss in the Prototypical Networks framework. A key idea is consistency learning: by augmenting unlabeled user utterances, PROTAUGMENT enforce a more robust text representation learning. Unfortunately, back-translation is a poor data augmentation strategy for short-texts: neural machine translation provides very similar (if not the same) sentences to the original ones, which hinders its ability to provide diverse augmentations (Section 5.3). Consequently, in this work, we transfer a denoising autoencoder pre-trained on the sequence-to-sequence task (Lewis et al., 2020) to the paraphrase generation task and then use it to generate paraphrases. As fine-tuning is very efficient for such a model, it is not easy to optimize it for diverse paraphrasing. (Goyal and Durrett, 2020) presents an approach for diverse paraphrasing that reorders the original sentence to guide the conditional language model to generate diverse sentences. The diversity in that work is provided by the reordering of the elements, which surprisingly affects the attention mechanism. In (Liu et al., 2020), expression diversity is part of the unsupervised paraphrasing system supported by simulated annealing. Both approaches imply domain transfer, and consequently, as many diverse paraphrasing models to maintain as the number of considered application domains, which do not scale very well. In this work, we instead introduce diversity in the downstream decoding algorithm used for paraphrase generation. Diverse decoding methods are mostly extensions to the beam search algorithm, including noise-based algorithms (Cho, 2016), iterative beam search (Kulikov et al., 2019), clustered beam search (Tam, 2020) and diverse beam search (Vijayakumar et al., 2018). There is no clear optimal solution, the choice is task-specific and dependent on one's tolerance for lower quality outputs as a diversity/fluency trade-off (Ippolito et al., 2019). While diverse beam search allows controlling the diversity/fluency trade-off partially, we further demonstrate that adding constraints to diverse beam search in order to generate tokens not seen in the input sentence (that is, *constrained diverse beam search*) is a simple yet powerful strategy to further improve the diversity of the paraphrases. Paired with paraphrasing user utterances and its consistency loss incorporated in Prototypical networks, our model is the best method for intent detection meta-learning on 4 public datasets, with neither extra labeling efforts nor domain-specific conditional language model fine-tuning. We also show that PROTAUGMENT, having access to only 10 samples of each class of the training data, still significantly outperforms a Prototypical Network which is given access to *all* samples of the same training data.

2 Neural architectures for meta-learning

Past works on meta-learning for classification tasks investigate how to best predict a query point's class at an episode scale. This process is bounded to the set of the C classes considered in a given episode. Matching Networks (Vinyals et al., 2016) predict the class of a query point as the average cosine distance between the query vector and all support vectors for each class. Prototypical Networks (Snell et al., 2017) extend Matching Networks: after obtaining support vectors from the encoder, a class *prototype* is produced via a class-wise vector averaging operation. All query points are then predicted with respect to their distance (cosine or euclidean) to all prototypes. Like Prototypical Networks, Relation Networks (Sung et al., 2018) emerged from Computer Vision application and were later successfully applied to NLP (Zhang et al., 2018). They introduce a relation module, which captures the relationship between data points: instead of using a pre-defined distance (euclidean or cosine most of the time), this approach allows such networks to learn this metric by themselves. This is achieved using either a shallow feed-forward sub-network or a Neural Tensor Layer relation module (Socher et al., 2013) (intermediate learnable matrices). Another extension to Prototypical Networks is provided in (Ren et al., 2018). Unlabeled data are incorporated using two distinct approaches: i) taking unlabeled data from the same classes as the episode or ii) using any unlabeled data and incorporating both a distractor cluster and masking strategy to minimize the impact of distant unlabeled points. The first approach is unrealistic for meta-learning, as it implies knowing the unlabeled data class. The second method assumes that all the noise is centered around a single distractor cluster and introduces an additional hyperparameter for masking – which is hardly fine-tuneable for small few-shot datasets.

3 Background

3.1 Notations

Meta-learning algorithms are trained using a specific procedure made of consecutive episodes. Let $\mathcal{C}_{ep}$ be the set of C classes sampled for the current training episode, such as $\mathcal{C}_{ep} \subset \mathcal{C}_{train}$, where $\mathcal{C}_{train}$ is the set of all classes available for training. We note $\mathcal{C}_{test}$, the set of classes used for testing, with $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \emptyset$. Each class $c \in \mathcal{C}_{ep}$ comes with K labeled samples, used as support. The set of $C \times K$ samples are usually referred to as $\mathcal{S}$, the support set, so that $\mathcal{S} = \{(x_1, y_1), \ldots, (x_{C\times K}, y_{C\times K})\}$. We denote S_c the set of support examples labeled with class c. Each episode comes with a query set $\mathcal{Q}$, which serves as the episode-scale optimization – the model parameters are updated based on the prediction loss on $\mathcal{Q}$, given $\mathcal{S}$ as an input. $\mathcal{Q}_c$ is the set of query examples labeled with class c.

3.2 Prototypical networks

In prototypical networks, each class is mapped to a representative point, called *prototype*. Each sample is first encoded into a vector using an embedding function f_ϕ with learnable parameters ϕ – this is the function we want to optimize. Using these embeddings, we compute each prototype $p_c, c \in \mathcal{C}_{ep}$ as the mean vector of embedded support points belonging to the class c, as described in Equation 1.

$$p_c = \frac{1}{K} \sum_{(x_i, y_i) \in S_c} f_\phi(x_i) \quad (1)$$

Given those prototypes and a distance function d, prototypical networks assign a label to a query point by computing the softmax over distances between this point's embedding and the prototypes, as in Equation 2. In the original paper, (Snell et al., 2017) use the euclidean distance and we also observed consistent slightly worse results with the cosine distance.

$$\mathbb{P}_\phi(y = c|x) = \text{softmax}\left(-d(f_\phi(x), p_c)\right) \quad (2)$$

The supervised loss function $\bar{L}$ is the average negative log-probability of the correct class assignments for all query points. At test time, episodes are created using classes from $\mathcal{C}_{test}$, and accuracy is measured as the query points assignments, given prototypes derived from the support points.

4 PROTAUGMENT

In this section, we present our semi-supervised approach PROTAUGMENT. Along with the labeled data randomly chosen at each episode, this approach uses U unlabeled data randomly drawn from the whole dataset – that is, data from training, validation, and test labels. We first do a data augmentation step from this unlabeled data, where we obtain M paraphrases for each unlabeled sentence. The m^{th} paraphrase of x will be denoted $\tilde{x}^m$. Then, given unlabeled data and their paraphrases, we compute a fully unsupervised loss. Finally, we combine both the supervised loss $\bar{L}$ (the Prototypical Network loss using labeled data) and unsupervised loss (denoted $\tilde{L}$) and run back-propagation to update the model's parameters.

4.1 Generating augmentations through paraphrasing

The BART (Lewis et al., 2020) model is a Transformer-based neural machine translation architecture that is trained to remove artificially corrupted text from the input thanks to an autoencoder architecture. While it is trained to reconstruct the original noised input, it can be fine-tuned for task-specific conditional generation by minimizing the cross-entropy loss on new training input-output pairs (Bevilacqua et al., 2020). In PROTAUGMENT, we fine-tune a pre-trained BART model on the paraphrasing task. The paraphrase sentence pairs we use for this task are taken from 3 different paraphrase detection datasets[1]: Quora (Sharma et al., 2019), MSR (Zhao and Wang, 2010), and Google PAWS-Wiki (Yang et al., 2019; Zhang et al., 2019). Those datasets have different sizes, and the largest one – Quora – consist of 149,263 pairs of duplicate questions. To balance turns of sentences (questions/non questions paraphrases), 50% of our fine-tuning paraphrase datasets is made of Quora, 5.6% of MSR and 44.4% PAWS-Wiki. This yields 94,702 sentence pairs to train the model on the paraphrasing task. We include both code and data on our github repository [2].

Using this fine-trained paraphrasing model, we can generate paraphrases of unlabeled sentences, hopefully having paraphrases representing the same intents as the original sentences. To add some diversity in the generated paraphrases, we use Di-

[1] we take only pairs that are paraphrases of each other since these are *paraphrase detection* datasets

[2] https://github.com/tdopierre/ProtAugment

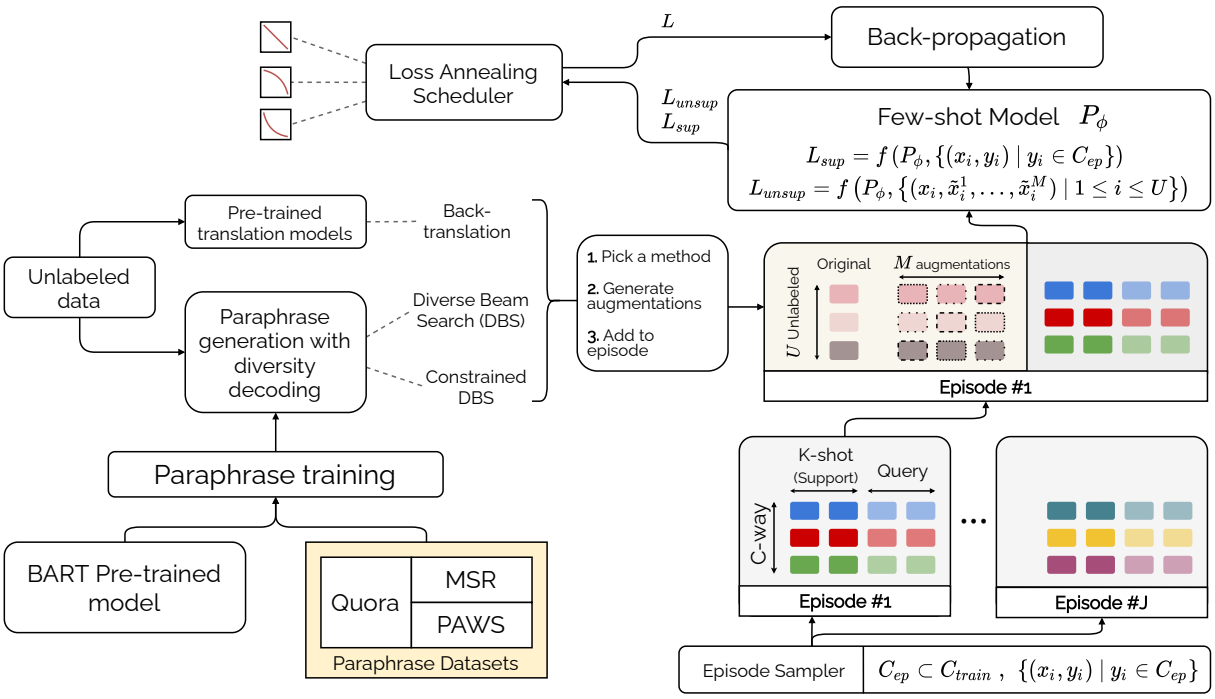

Figure 1: PROTAUGMENT illustrated on a 3-way 2-shot short text classification meta-learning task ($C = 3$, $K = 2$). BART is pre-trained for the paraphrasing task on three datasets: Quora (Sharma et al., 2019), MSR (Zhao and Wang, 2010) and Google PAWS-Wiki (Yang et al., 2019; Zhang et al., 2019). The paraphrase model is used to paraphrase unlabeled samples but equipped with diversity strategies (back translation being proposed as a baseline). The final loss is computed using a loss annealing scheduler, which is expected to smooth the supervised (given shots) and unsupervised (augmented unlabeled sentences) prediction errors to yield parameter gradients. A new episode means sampling other classes along with their support and query points.

verse Beam Search (DBS) instead of the regular Beam Search. As Vijayakumar et al. (2018) has shown in the original paper, adding a dissimilarity term during the decoding step helps the model produce sequences that are quite far from each other while still retaining the same meaning. The next section describes how we constrained this decoding to enforce even more diversity among generated paraphrases in PROTAUGMENT.

4.2 Constrained user utterances generation

While DBS enforces diversity between the generated sentences, it does not ensure diversity between the generated paraphrases and the original sentences. It was formerly designed for tasks that do not need this diversity with the original sentence (translation, image captioning, question generation). To enforce that our generated paraphrases are diverse enough, we further constraint DBS by forbidding using parts of the original sentences. In the following paragraphs, we introduce two forbidding strategies.

Unigram Masking. In this strategy, we randomly select tokens from the input sentence which will be forbidden at the generation step. The goal here is to force the model to use different words in the generated sentences than it saw in the original sentences. Each word of the input sentence is randomly masked using a probability p_{mask}. The underlying assumption is that forbidding tokens at the beginning of a sentence with a higher probability than the end of the sentence may have a greater impact on the beam search algorithm. Indeed, as the decoding is a conditional task based on prior generated tokens, masking the first tokens may significantly impact diversity. We therefore introduce two additional variants: one where we put more probability on the first tokens and the reverse where there is more weight in the last tokens. To ensure that all three variants mask the same amount of tokens on average, we ensure the area under the curve of the three probability functions are equal to a fixed value noted p_{mask}.

Bi-gram Masking Another strategy we consider is to prevent the paraphrasing model from generating the same bi-grams as in the original sentence. This time, we are not masking any single word but

forcing the model to change the sentence's structure, which will, hopefully, increase the diversity of the generated paraphrases.

4.3 Unsupervised diverse paraphrasing loss

After generating paraphrases for each unlabeled sentence, we create unlabeled prototypes. For each unlabeled sentence $x_u \in U$, we derive the unlabeled prototype p_{x_u} as the average embedding of the paraphrases of x_u (Equation 3).

$$p_{x_u} = \frac{1}{M} \sum_{m=1}^{M} f_\phi(\tilde{x}_u^m) \quad (3)$$

After obtaining the unlabeled prototypes, we compute the distances between all unlabeled samples and all unlabeled prototypes. Given such distances, we model the probability of each unlabeled sample being assigned to each unlabeled prototype (Equation 4), as in the supervised part of the Prototypical Networks – except this time, it is fully unsupervised. This probability should be close to 1 between an unlabeled sample and its associated unlabeled prototype and close to 0 otherwise.

$$\mathbb{P}_\phi(u = v|x_u) = \text{softmax}\left(-d(f_\phi(x_u), p_{x_v})\right) \quad (4)$$

Given assign probabilities between unlabeled samples and unlabeled prototypes, we can compute a fully unsupervised cross-entropy loss $\tilde{L}$, training the model to bring each sentence closer to its augmentations' prototype and further from the prototypes of other unlabeled sentences. Recall that f_ϕ is the embedding function with ϕ as learnable parameters (Section 3.2).

After obtaining both supervised loss $\bar{L}$ and unsupervised loss $\tilde{L}$, we combine them into the final loss L using a loss annealing scheduler (see Equation 5), which will gradually incorporate the unsupervised loss as training progresses.

$$L = t^\alpha \times \tilde{L} + (1 - t^\alpha) \times \bar{L} \quad ; \quad t \in (0, 1) \quad (5)$$

The goal here is to mainly use the supervised loss first so that the model gets a sense of the classification task. Then, incorporating more and more knowledge from unlabeled samples will make the model more robust to noise, which is essential as it is constantly tested on classes it has never seen before. We explore three different strategies for gradually increasing the unsupervised contribution: a linear approach ($\alpha = 1$), an aggressive one ($\alpha = 0.25$), and a conservative one ($\alpha = 4$).

5 Experiments

5.1 Datasets

We consider the DialoGLUE benchmark (Mehri et al., 2020), a set of natural language understanding benchmark for task-oriented dialogue, which contains three datasets for intent detection: `Banking77`, `HWU64` and `Clinic150` – the three datasets were already available prior the release of DialoGLUE. Additionally, we also consider the `Liu57` intent detection dataset, as it contains the same order of magnitude of intent classes and is user-generated as well. All datasets are public and in English.

Banking77 The `Banking77` dataset (Casanueva et al., 2020) classifies 13,083 user utterances related to into 77 different intents. This dataset i) is specific to a single domain (banking) and ii) requires a fine-grained understanding to classify due to intents being very similar. Following (Mehri et al., 2020) and contrary to (Casanueva et al., 2020), we designate a validation set along a training and a testing set for that dataset (Table 1).

HWU64 `HWU64` (Xingkun Liu and Rieser, 2019) classifies 25,716 user utterances with 64 user intents. It features intents spanning across 21 domains (alarm, audio, audiobook, calendar, cooking, datetime, ...). When separating training, validation, and test labels, we ensure each domain is rep-

Dataset	#sentences	#classes train/valid/test (total)	Available sentences/class	#tokens/sentence
Banking77	13,083	25/25/27(77)	170 ± 33	11.7 ± 7.6
HWU64	11,036	23/16.4/24.6(64)	172 ± 40	6.6 ± 2.9
Clinic150	22,500	50/50/50(150)	150 ± 0	8.5 ± 3.3
Liu	25,478	18/18/18(54)	472 ± 831	7.5 ± 3.4

Table 1: Main statistics of intent detection evaluation datasets. For HWU64, each split's number of classes varies at each run to ensure there is no cross-split domain, hence the decimal number.

resented only in one set of labels. This ensures the model learns to discriminate between both intents and domains.

Clinic150 This dataset (Larson et al., 2019) classifies 150 user intents in perfectly equally-distributed classes. This chatbot-like style dataset was initially designed to detect out-of-scope queries, though, in our experiments, we discard the out-of-scope class and only keep the 150 labeled classes to work with, as in (Mehri et al., 2020).

Liu57 Introduced by Liu et al. (2019a), this intent detection dataset is composed of 54 classes. It was collected on Amazon Mechanical Turk, where workers were asked to formulate queries for a given intent with their own words. It is highly imbalanced: the most (resp. least) common class holds $5,920$ (resp. 24) samples

5.2 Experimental settings

Conditional language model and language model. For the BART fine-tuning process, we used the defaults hyper-parameters reported in (Lewis et al., 2020), and we fine-tuned the BART model for a single epoch (two hours on a Titan RTX GPU). Increasing the number of epochs for fine-tuning BART degrades performances on the intent detection task: the downstream diverse beam search struggles to find diverse enough beam groups since the model perplexity has been lower with further fine-tuning (this is also hinted in (Bevilacqua et al., 2020)). Our text encoder f_ϕ is a `bert-base` model, and the embedding of a given sentence is the last layer hidden state of the first token of this sentence. For each dataset, this model is fine-tuned on the masked language modeling task for 20 epochs. Then, the encoder of our meta learner is initialized using the weights of this fine-tuned model.

Datasets From a dataset point-of-view, we create two data profiles: **full** (all the training dataset is available, the usual meta-learning scenario) and **low** (only 10 samples are available for each training class, an even more challenging meta-learning scenario in which a model meta-learns on very few samples per training class). All experimental setups are run 5 times. For each run, we randomly select training, validation, and testing classes, as well as the samples for the **low** setting. We train the few-shot models for a maximum of $10,000$ C-way K-shots episodes, evaluating and testing every 100 episodes, stopping early if the evaluation accuracy has not progressed for at least 20 evaluations. We evaluate and test using 600 episodes, as in other few-shot works (Snell et al., 2017; Chen et al., 2019). We compare the systems in the following standard few-shot evaluation scenarios: 5-way 1-shot, and 5-way 5-shots.

Paraphrasing. At each episode, we draw $U = 5$ unlabeled samples to generate paraphrases from. For the back-translation baseline, we use the publicly available[3] translation models from the `Helsinki-NLP` team. We use the following pivot languages: `fr`, `es`, `it`, `de`, `nl`, which yields 5 augmentations for each unlabeled sentence. For our experiments with Diverse Beam Search, we generate sentences using 15 beams, group them into 5 groups of 3 beams. In each group, we select the generated sentence which is the most different from the input sentence using BLEU as a metric for diversity. This yields $M = 5$ paraphrases for each unlabeled sentence, as in the back-translation baseline. DBS uses a diversity penalty parameter to penalize words that have already been generated by other beams to enforce diversity. As advised in the original DBS paper (Vijayakumar et al., 2018), we set the diversity penalty to 0.5 in our experiments, which provides diversity while limiting model hallucinations. Our Unigram Masking strategy's masking probability is set to $p_{\text{mask}} = 0.7$ found by linear search from 0 to 1 with steps of 0.1.

orig: How long will my transfer be pending for? *back*: How long will my transfer be on hold? *dbs_0*: How long will my transfer be pending? I am in first year. *dsb_1*: When are all transfers coming up and how many days are they expected? *dbs_2*: If I have a transfer for a while, how long should I wait for it?
orig: I am not sure where my phone is. *back*: I don't know where my phone is. *dbs_0*: I am not really sure where my phone is located *dsb_1*: How can I find the location of any Android mobile *dbs_2*: I don't know where is my cell phone
orig: can you play m3 file *back*: can you read m3 file *dbs_0*: M3 files: can I play the entire M3 file? *dsb_1*: Is there any way to play 3M files on Earth without downloading it *dbs_2*: Is there any way to play M3 files on Windows?

Table 2: Examples of sentences (*orig*) paraphrased using back translation (*back*), vanilla diverse beam search – DBS (*dbs_0*), DBS with unigram masking (*dbs_1*) and DBS with bigram masking (*dbs_2*)..

5.3 Evaluation of paraphrase diversity

We evaluate the diversity of paraphrases for each method, and report results for two representative datasets in Table 3 (due to space limitations, the

[3]https://huggingface.co/models?search=helsinki-nlp

report for all datasets is given in appendix B). For each paraphrasing method and each dataset, metrics are computed over unlabeled sentences and their paraphrases. To assess the diversity of paraphrases generated by the different methods, the popular BLEU metric in Neural Machine Translation is a poor choice (Bawden et al., 2020). We use the bi-gram diversity (**dist-2**) metric as proposed by (Ippolito et al., 2019), which computes the number of distinct 2-grams divided by the total amount of tokens. We also report the average similarity (denoted **use**) within each sentence set, using the Universal Sentence Encoder as an independent sentence encoder. Results show that paraphrases obtained with back-translation are too close to each other, resulting in a high sentence similarity and low bi-gram diversity. On the other hand, DBS generates more diverse sentences with a lower similarity. Our masking strategies strengthen this effect and yield even more diversity. The measured diversity strongly correlates with the average accuracy of the intent detection task (Table 4).

	BANKING77		HWU64	
	dist-2	**use**	**dist-2**	**use**
back-translation	0.183	0.896	0.307	0.888
DBS	0.200	0.807	0.340	0.769
DBS+bigram	0.228	0.702	0.350	0.692
DBS+unigram	**0.343**	**0.613**	**0.407**	**0.628**

Table 3: Paraphrase diversity measures. For **dist-2** (resp. **use**) higher values (resp. lower) indicates more diversity.

5.4 Intent detection results

In this section, we discuss the accuracy results for the different meta-learners, for the standard 5-way and $\{1, 5\}$-shots meta-learning scenarios, as provided in Table 4. The reported metric is the accuracy on the test set at the iteration where the validation set's accuracy is maximal. Our DBS+unigram strategy row corresponds to the `flat` masking strategy, with $p_{\text{mask}} = 0.7$. First, all methods augmented with unsupervised diverse paraphrasing outperform prototypical networks. However, back translation demonstrates only a limited improvement over the vanilla prototypical network due to their narrow diversity for short texts. Using paraphrases from DBS yields better results – about 0.5 points over BT, on average –, hinting that using diverse paraphrases in the unsupervised consistency loss allows the few-shot model to build more robust sentence representations and therefore provides improved generalization capacities. Those results are consistent across the different datasets, except for Clinic for which accuracies are all very high, making all methods hardly separable. The dataset is not challenging enough, or in other words, meta-learning is robust to unbalanced short text classification problems given the nature of that dataset.

These results illustrate the need for unsupervised paraphrasing and show that using diverse paraphrases provide a significant performance leap. In the 1-shot (resp. 5-shot) scenario, our best meta-learner improves prototypical networks by 5.27 (resp. 2.85) points on average. Remember that these improvements are made in an unsupervised manner hence at no additional cost. Slightly different from to (Xie et al., 2020), we do not find statistical differences depending on the rate at which $\tilde{L}$ is annealed in PROTAUGMENT loss ($\alpha \in \{0.25, 1, 4\}$), which makes it easier to tune – our unsupervised loss serves as a consistency regularization. Due to space limitations, this analysis is available in appendix D.

Adding our masking strategies on top of DBS has a significant impact on all datasets, with the unigram variant being up about 2 points over the vanilla DBS on average. On all datasets except Clinic, given only 10 labeled samples per class (**low** profile), it even outperforms the supervised baseline which is given the full training data (**full** profile). This means that PROTAUGMENT does better than prototypical networks with much less – 15 times, and up to 47 times, depending on the dataset – labeled sentences per class. Those results indicate that our method more than compensates for the lack of labeled data and that no matter the amount of data available for the training class, there is a performance ceiling you cannot overcome without adding unsupervised knowledge from the validation and test classes. In the **full** profile, when given all the training data, our method greatly surpasses the Prototypical Network – 3.58 points given 1 shot, on average. Moreover, PROTAUGMENT is not only suited for the case where very little training data is available (**low** profile): when sampling shots from the entire training dataset (**full** profile), it outperforms a fully supervised baseline. Furthermore, note that our method is consistently more stable than the supervised baselines, as its average standard deviation over the different runs is much lower than the vanilla Prototypical Network.

Data Profile	Method	Datasets								Accuracy stats ($AVG \pm STD$)	
		Banking		HWU		Liu		Clinic			
		$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$
low profile	Prototypical Network	82.20	91.57	74.37	86.48	80.06	89.62	94.29	98.10	82.73 ± 2.32	91.44 ± 1.92
	ours w/ BT	83.83	92.16	78.70	89.36	80.84	90.87	94.06	97.62	84.36 ± 1.15	92 .50 ± 0.94
	ours w/ DBS	83.10	92.56	80.06	90.21	82.31	91.64	93.70	97.83	84.80 ± 1.26	93.06 ± 0.99
	ours w/ DBS+bigram	86.04	93.55	82.09	**91.57**	83.60	92.71	95.11	98.23	86.71 ± 1.14	94.01 ± 1.05
	ours w/ DBS+unigram	**87.23**	**94.29**	**83.70**	91.29	**85.16**	**93.00**	**95.92**	**98.56**	**88.00 ± 1.22**	**94.29 ± 0.76**
full profile	Prototypical Network	86.28	93.94	77.09	89.02	82.76	91.37	96.05	98.61	85.55 ± 2.20	93.24 ± 1.22
	ours w/ BT	87.46	94.47	81.31	91.44	84.14	92.67	95.19	98.36	87.02 ± 1.36	94.23 ± 0.82
	ours w/ DBS	86.94	94.50	82.35	91.68	84.42	92.62	94.85	98.41	87.14 ± 1.36	94.30 ± 0.60
	ours w/ DBS+bigram	88.14	94.70	84.05	92.14	85.29	93.23	95.77	98.50	88.31 ± 1.43	94.64 ± 0.59
	ours w/ DBS+unigram	**89.56**	**94.71**	**84.34**	**92.55**	**86.11**	**93.70**	**96.49**	**98.74**	**89.13 ± 1.13**	**94.92 ± 0.57**

Table 4: 5-way 1-shots and 5-way 5-shots accuracy on the test sets for each dataset. The *ours* method is PROTAUGMENT (unsupervised consistency loss using diverse paraphrases) equipped with different paraphrasing strategies. For each dataset × C-way K-shot setting, we compute the average and the standard deviation over the 5 runs (see Section 5.2), so that the last two columns contains average accuracy and ± the average standard deviations. For each data profile, we highlight the best method in **bold**. We underline the methods on the **low** profile which perform better than the Prototypical Networks on the **full** profile. We trained 400 different meta-learners – 5 methods, 2 data profiles, 4 datasets, 2 meta-learning setup ($K = 1, 5$) and 5 runs for each configuration.

5.5 Masking strategies

We experimented with three variants of the unigram strategy (Section 4.2), each assigning a different drop chance to each token depending on its position in the input sentence. In our experiments, we did not observe any significant difference in performance when putting more weight on the first tokens (*down*), or last tokens (*up*), or the same weight on all tokens (*flat*) (Detailed results in appendix C). We also conducted experiments where we tune the value p_{mask}, from 0 to 1, selecting 0.7 as the best trade-off (Figure 2). This figure also clearly shows that the *Clinic* dataset is one order of magnitude easier to solve than the other datasets.

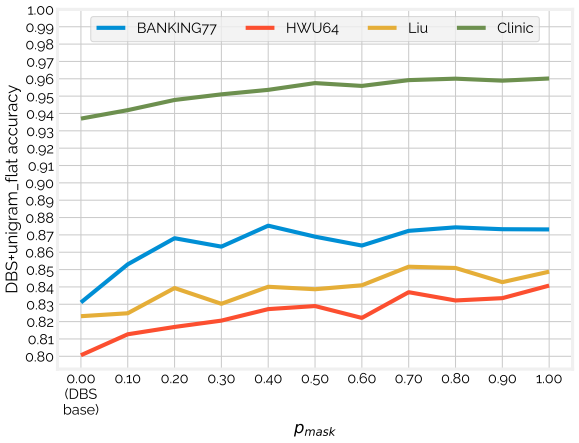

Figure 2: 5-way 1-shot accuracy of DBS-unigram-flat method using different values of p_{mask}. Setting this value to 0 corresponds to the vanilla DBS without masking strategies.

6 Conclusion

In this work, we proposed PROTAUGMENT, an architecture for meta-learning for the problem of classifying user-generated short-texts (intents). We first introduced an unsupervised paraphrasing consistency loss in the prototypical network's framework to improve its representational power. Then, while the recent diverse beam search algorithm was designed to enforce diversity between the generated paraphrases, it does not ensure diversity between the generated paraphrases and the original sentences. To make up for the latter, we introduce constraints in the diverse beam search generation, further increasing the diversity. Our thorough evaluation demonstrates that PROTAUGMENT offers a significant leap in accuracy for the most recent and challenging datasets. PROTAUGMENT vastly outperforms prototypical networks, which was found to be the best meta-learning framework for short-texts (Dopierre et al., 2021) against unsupervised-extended Prototypical Networks (Ren et al., 2018), Matching Networks (Vinyals et al., 2016), Relation Networks (Sung et al., 2018), and Induction Networks (Geng et al., 2019), thereby making PROTAUGMENT the new state-of-the-art for this task. We provide the source code of PROTAUGMENT as well as code for evaluations reported in this paper on a public repository [4]

[4] https://github.com/tdopierre/ProtAugment

Acknowledgments

We are thankful for the discussion we had with Michele Bevilacqua, Marco Maru, and Roberto Navigli from Sapienza university about diversity in Natural Language Generation. We also would like to thank ANRT [5] for making partnerships between companies and universities happen.

References

Rachel Bawden, Biao Zhang, Lisa Yankovskaya, Andre Tättar, and Matt Post. 2020. A study in improving BLEU reference coverage with diverse automatic paraphrasing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 918–932, Online. Association for Computational Linguistics.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221, Online. Association for Computational Linguistics.

Xu Cao, Deyi Xiong, Chongyang Shi, Chao Wang, Yao Meng, and Changjian Hu. 2020. Balanced joint adversarial training for robust intent detection and slot filling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4926–4936.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. In *International Conference on Learning Representations*.

Kyunghyun Cho. 2016. Noisy parallel approximate decoding for conditional recurrent language model. *arXiv preprint arXiv:1605.03835*.

Alice Coucke, Alaa Saade, Adrien Ball, and Bluche et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Thomas Dopierre, Christophe Gravier, and Thomas Logerais. 2021. A neural few-shot text classification reality check. In *Proc. of EACL 2021*.

Thomas Dopierre, Christophe Gravier, Julien Subercaze, and Wilfried Logerais. 2020. Few-shot pseudo-labeling for intent detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4993–5003, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197, Hong Kong, China. Association for Computational Linguistics.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3904–3913, Hong Kong, China. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252, Online. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2020. Investigating learning dynamics of BERT fine-tuning. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 87–92, Suzhou, China. Association for Computational Linguistics.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana. Association for Computational Linguistics.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762, Florence, Italy. Association for Computational Linguistics.

Ashish Khetan and Zohar Karnin. 2020. schuBERT: Optimizing elements of BERT. In *Proceedings of the*

[5] https://www.anrt.asso.fr/fr

58th Annual Meeting of the Association for Computational Linguistics, pages 2807–2818, Online. Association for Computational Linguistics.

Ilia Kulikov, Alexander Miller, Kyunghyun Cho, and Jason Weston. 2019. Importance of search and evaluation strategies in neural dialogue modeling. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 76–87, Tokyo, Japan. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online. Association for Computational Linguistics.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019a. Benchmarking natural language understanding services for building conversational agents. *arXiv preprint arXiv:1903.05566*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. Association for Computational Linguistics.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. 2018. Meta-learning for semi-supervised few-shot classification. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci. 2019. Natural language understanding with the quora question pairs dataset.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yik-Cheung Tam. 2020. Cluster-based beam search for pointer-generator chatbot grounded by knowledge. *Computer Speech & Language*, 64:101094.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 7371–7379. AAAI Press.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Qizhe Xie, Zihang Dai, Eduard H. Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Pawel Swietojanski Xingkun Liu, Arash Eshghi and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages xxx–xxx, Ortigia, Siracusa (SR), Italy. Springer.

Shuo Yang, Lu Liu, and Min Xu. 2021. Free lunch for few-shot learning: Distribution calibration. In *Proceedings of the 9th International Conference on Learning Representations*, page (Accepted paper to appear), Online. OpenReview.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods*

in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Changlong Yu, Jialong Han, Haisong Zhang, and Wilfred Ng. 2020. Hypernymy detection for low-resource languages via meta learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3656, Online. Association for Computational Linguistics.

Wen Zhang, Jiawei Hu, Yang Feng, and Qun Liu. 2018. Refining source representations with relation networks for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1292–1303, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Shiqi Zhao and Haifeng Wang. 2010. Paraphrases and applications. In *Coling 2010: Paraphrases and Applications–Tutorial notes*, pages 1–87, Beijing, China. Coling 2010 Organizing Committee.

A Diverse paraphrase samples

orig: Are you considered a bot? *back*: Do you consider yourself a bot? *dbs_0*: Is there any way to be considered a bot? *dsb_1*: Is there any way to know if someone is bot? *dbs_2*: Is there any way to know if a person is bot?
orig: what is the square root of 31 *back*: what is the square root of 31 *dbs_0*: How can I find square roots of 31 in mathematics? *dsb_1*: How can I find quadratic roots of 31? *dbs_2*: What is square roots and what is its effect on 31
orig: How long will my transfer be pending for? *back*: How long will my transfer be on hold? *dbs_0*: How long will my transfer be pending? I am in first year. *dsb_1*: When are all transfers coming up and how many days are they expected? *dbs_2*: If I have a transfer for a while, how long should I wait for it?
orig: How to cancel a transaction *back*: How to cancel a transaction *dbs_0*: What are some ways to cancel a transaction (in any way)? *dsb_1*: What are some ways of cancelling an account (in some cases also involving transaction). *dbs_2*: Is there any way in which I can cancel an existing transaction?
orig: I am not sure where my phone is. *back*: I don't know where my phone is. *dbs_0*: I am not really sure where my phone is located *dsb_1*: How can I find the location of any Android mobile *dbs_2*: I don't know where is my cell phone
orig: What do I need to do for a refund? *back*: What do I need to do to get a refund? *dbs_0*: What should I do now that I am not getting a refund? *dsb_1*: What are things that should be done before resumption of service (reimbursed)? *dbs_2*: What should you do if you want to refund your period of data?
orig: does pizza hut have my order ready *back*: Does Pizza Hut has my order ready *dbs_0*: Does the pizza Hut have all your orders ready? *dsb_1*: Does pizza Hut have all your orders made up? *dbs_2*: Does the pizza Hut have all your orders ready for delivery?
orig: go silent for a day *back*: Quiet for a day. *dbs_0*: Do you stay silent for a day or go silent for another day *dsb_1*: To the rest of the day, stay peaceful and collected. *dbs_2*: So, to the rest of the day, go silent only.
orig: what's the recipe for fish soup *back*: What is the recipe for fish soup *dbs_0*: How do you make fish soup? How is the recipe determined? *dsb_1*: How can you recipe for fish-sugary food? *dbs_2*: What are the recipes for Fish soup and how is it prepared?
orig: Find easy recipe for almond milk *back*: Find an easy recipe for almond milk *dbs_0*: What are some good recipe for Almond milk? *dsb_1*: What are some good ways of making Almond milk? *dbs_2*: How do I make Almond milk for a beginner?
orig: Will I need to wear a coat today? *back*: Should I wear a coat today? *dbs_0*: Today, do I need to put on a coat *dsb_1*: Should I wear a coat and what kind of coat *dbs_2*: What should I wear to work today, and why
orig: can you play m3 file *back*: can you read m3 file *dbs_0*: M3 files: can I play the entire M3 file? *dsb_1*: Is there any way to play 3M files on Earth without downloading it *dbs_2*: Is there any way to play M3 files on Windows?

Table 5: Additional paraphrases samples.

B Paraphrase Diversity Evaluation

	BANKING77			HWU64			Liu			Clinic		
	BLEU	**dist-2**	**use**	**BLEU**	**dist-2**	**use**	**BLEU**	**dist-2**	**use**	**BLEU**	**dist-2**	**use**
back-translation	56.0	0.183	0.896	40.2	0.307	0.888	47.7	0.268	0.892	43.9	0.205	0.903
DBS	34.2	0.200	0.807	19.5	0.340	0.769	19.7	0.293	0.750	22.3	0.236	0.805
DBS+bigram	0.1	0.228	0.702	0.1	0.350	0.692	0.4	0.293	0.664	0.2	0.257	0.717
DBS+unigram	0.2	0.343	0.613	0.5	0.407	0.628	0.5	0.351	0.596	0.3	0.323	0.644

Table 6: Paraphrase evaluation on all 4 datasets. The unigram variant exposed here is using the *flat* masking strategy with $p_{\text{mask}} = 0.7$.

C Masking tokens depending on their position

Method	**Datasets**								**Accuracy stats**	
	Banking		HWU		Liu		Clinic		$(AVG \pm STD)$	
	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$
DBS+unigram-*flat*	87.23	94.29	83.70	91.29	85.16	93.00	95.92	98.56	88.00 ± 1.22	94.29 ± 0.76
DBS+unigram-*down*	87.43	94.14	83.06	92.14	84.87	93.33	95.93	98.61	87.82 ± 0.84	94.55 ± 0.71
DBS+unigram-*up*	86.18	94.12	83.30	91.21	85.14	93.15	95.84	98.30	87.62 ± 1.23	94.20 ± 0.70

Table 7: Performances of DBS+unigram strategies putting either more chance to mask first tokens (*down*), last tokens (*up*), or the same chance to all tokens (*flat*). All strategies use $p_{\text{mask}} = 0.7$. Overall, there is no significant difference between the three strategies.

D Loss annealing strategy

Method		**Datasets**								**Accuracy stats**	
		Banking		HWU		Liu		Clinic		$(AVG \pm STD)$	
	α	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$	$K=1$	$K=5$
DBS+unigram-*flat*	1	87.23	94.29	83.70	91.29	85.16	93.00	95.92	98.56	88.00 ± 1.22	94.29 ± 0.76
	0.25	86.71	94.17	82.71	91.19	85.52	93.11	95.99	98.44	87.73 ± 1.09	94.23 ± 0.85
	4	86.90	94.14	83.26	92.35	84.48	93.17	95.69	98.49	87.58 ± 1.64	94.54 ± 0.81

Table 8: Performances of DBS+unigram strategies with different values of the loss annealing parameter α. All strategies use $p_{\text{mask}} = 0.7$. Overall, there is no significant difference when changing the value of α.