Exploring the Dimensions of Few-Shot Named Entity Recognition

Chandan Akiti, Murali Nandan Nagarapu, Sai Ajay Modukuri
Pennsylvania State University
{cra5302, mzn5322, svm6822}@psu.edu

Abstract

We present an empirical analysis of Few-Shot Learning for Named Entity Recognition task. We use Prototypical Networks as the base model and explore the dimensions of few-shot learning. Specifically, we experiment with two types of distance metrics, output dimension sizes, layer-wise BERT training and domain-transfer settings. Our results show the correlation of the choice of distance metric and the model hyper-parameters with the performance on test episodes. Using the insights from our analysis, we achieve SOTA for Few-Shot Learning on OntoNotes 5.0 dataset. We perform the experiments on OntoNotes dataset and show domain-transfer ability with I2B2 and WNUT NER datasets.

1 Introduction

Named Entity Recognition (NER) aims at identifying spans of text related to a set of entity classes. Adapting state-of-the-art NER models to new domains is challenging as they have different/new entities. Few-shot NER methods adopt prior-art NER models to these new domains. We aim to focus our research on three spaces of few-shot NER task (1) Metric Learning, (2) BERTology of few-shot NER, and (3) meta-learning.

2 Related Work

Meta and Few-Shot Learning: Existing state-of-the-art few-shot learning methods (Snell et al., 2017; Han et al., 2018) proposed prototypical networks that learn to represent each semantic class by a prototype based on the labeled examples in its support set. Yang and Katiyar (2020) show that contextual representation space provided by language models have a better performance for NER task than the prototypical networks. Meta-Learning is a paradigm that draws on previously seen examples in order to learn and adapt to newer tasks. Our code [1] is based on Holla et al. (2020) that conducted meta-learning on word sense disambiguation task.

BERTology: Tenney et al. (2019) shows that BERT (Devlin et al., 2019) contains elements of the natural language processing pipeline: POS tagging, parsing, NER, semantic roles, and coreference. Understanding what is known in terms of how BERT is able to achieve state-of-the-art performances on the standard NLP tasks is heavily researched. A detailed survey of training objectives, modifications to the architectures, and over-parameterization issues have been studied in (Rogers et al., 2020)

Distance Metric While Snell et al. (2017) used Euclidean distance metric for prototypical networks, recent work by Wohlwend et al. (2019) explored hyperbolic geometry for the prototypical network for the text classification task. This paper shows the converge guarantees of the hyperbolic distance metric based prototypical network both theoretically and empirically. Motivated by this, we also perform experiments for determining the effects of the choice of the distance metric on prototypical few-shot learning.

3 Task and Dataset

We treat NER as a Few-Shot word-level classification problem. As the entity recognition is context-dependent, we cannot use individual words for entity decoding. Instead, we use the entire sentences tagged with entities as training and validation examples where we sample the sentences to simulate the N-way K-shot setting.

Dataset: For the experiments for our model, we use the standard NER datasets in three domains which stand the largest annotated datasets. These are OntoNotes 5.0 for the general domain, I2B2

[1] https://github.com/chandan047/MetaLearningForNER

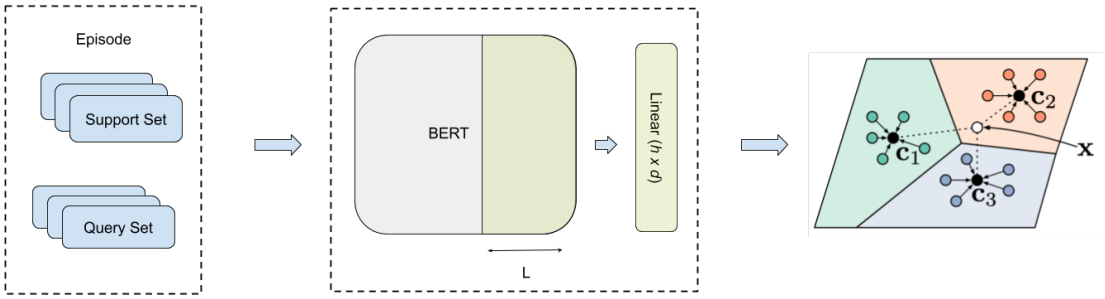


Figure 1: ProtoNet training pipeline indicating the trainable parameters in *green*. Each episode contains support and query sentences sampled greedily to include atleast K instances of each entity class. The contextual word representations are obtained from the BERT model after projecting it to a d-dimensional space. L represnts number of trainable layers in BERT. Finally, query instances are classified via Softmax over negative distances to class prototypes made from support instances.

Dataset	Domain	#Class	#Sent	#Entity
OntoNotes	General	18	76,714	104,151
I2B2'14	Medical	23	140,817	29,233
WNUT 17	Social	6	5,690	3,890

Table 1: Dataset Distribution

2014 for Medical, WNUT 2017 for Social domains. Another domain of news dataset, CoNLL 2003 was not considered due to the fewer classes in it. Diverse and representative labeling has been done for these datasets. For the train/val/test splits, we use the OntoNotes dataset with the baseline of Struct-Shot (Yang and Katiyar, 2020). The details of the dataset class distribution are in the table 1.

In our setting, we sample N-way K-shot episodes from the NER dataset. For each episode, we sample N entity classes greedily from the entities in the dataset such that least occuring entities appear more often in the episodes. For each entity, we sample sentences such that the sampled sentences have atleast K entity mentions in both support set and the query set. We choose the number of episodes per epoch such that median cardinality for the entities in the dataset is convered across the sampled episodes in that epoch. In this paper, we only focus on *6*-way *5*-shot experiments.

Training episodes: The training episodes are sampled from the **support domain**. We choose the support domain as OntoNotes since it is a general domain dataset. Following the experiments of (Yang and Katiyar, 2020), we choose the label set that has the worst performance with Few-Shot Nearest Neighbor classification. For all the experiments other than domain transfer, we chose a specific set of 12 labels for the training episode generation.

- `GPE`, `CARDINAL`, `PERCENT`, `TIME`, `LOC`, `EVENT`, `LANGUAGE`, `PERSON`, `DATE`, `FAC`, `MONEY`, `PRODUCT`, `O`

For the domain transfer experiments, we choose all the 18 labels (+O tag) for the train episodes.

Evaluation episodes: For the meta-evaluation and meta-test episodes, we chose a target domain and sample episodes for the entities in that domain. For evaluation we choose 30 episodes for the OntoNotes, 10 episodes for the I2B2 dataset, and similarly, ten episodes for the WNUT dataset.

4 Methods

Few-Shot classification for NER is shown to be effective on general domain data even without fine-tuning the Few-Shot classifiers. Yang and Katiyar (2020) use a supervised NER model fine-tuned on a general domain and apply Few-Shot classifier to decode the entity queries directly without Few-Shto specific training. Morever, they perform Viterbi decoding without CRF training. This shows that the prior NER models are very good few-shot learners for NER. In our setup, we describe the few-shot training method and analyze the effect of few-shot training on the Few-Shot NER classification task.

4.1 Model Architecture

We use $BERT_{BASE}$ model with 12 layers with L layers trainable and hidden size of 768. We have an additional linear layer with an output dimension d. Our prototypical networks use the d-dimensional contextual feature representations, and few-shot classify the query tokens using a distance metric as shown in the Figure 1.

4.2 Prototypical Networks

Our model **ProtoNet** is tightly based on Snell et al. (2017). A base model f_θ parameterized by θ is used to produce the prototype vector for every class as a mean vector μ_c, $c \in C$ of the representations of all the support data points for that class.

Given a distance function D defined on the representation space, the distribution over classes for a query is calculated as a softmax over negative distances to that class prototypes. We take support representations per NER entity class from f_θ using the tagged token representations in the support set. The distribution over classes for each query token is extracted as described above. The loss $\mathcal{L}^q$ is taken from this class distributions and then used for training the parameters θ.

Essentially, we are trying to minimize the following objective function

$$\mathbb{E}\left[\mathcal{L}_{P^s|s\sim\mathcal{S},q\sim\mathcal{Q}} \min_C D(P_c^s, q_c)\right] \quad (1)$$

where s, q are the support set and query set sampled i.i.d from uniformly sampled set of support sets S and query sets Q respectively. q_c is the query token of entity class c in the query set.

4.3 Model-Agnostic Meta-Learning

Model-Agnostic Meta-Learning (Proto-MAML) setting is an extended version of prototypical networks. The training paradigm of MAML consists of meta-episodes. Each meta-episode consists of training episodes sampled uniformly. MAML contains additional meta-parameters ϕ along with the base model parameters θ. ϕ are trained from scratch for each of the meta-episode and θ are updated after every meta-episode. Thus, MAML gives the ability to the meta-learning setting to learn the distance metric D for every meta-episode.

In our experiment, ϕ is used to classify the tokens to entities in the episode. This is implemented using the *higher* meta-learning framework from Grefenstette et al. (2019).

4.4 Few-Shot Nearest Neighbor Classifier

Similar to Yang and Katiyar (2020), our few-shot model uses a pre-trained NER base model. The query point is assigned the same class as the class of the closest support point. We fine-tune the model on the general domain NER dataset before few-shot inference. We obtain the nearest support neighbor using the cosine distance metric and asign the class for the support point.

4.5 Hyperparameters

We use a learning rate of *1e-5* and Adam optimizer in all our experiments. We explore the limits of Few-Shot learning by varying hyperparameters d and L for our ProtoNet model. Furthermore, we train for ten epochs or until early stopping and early stopping is employed when the validation does not improve for two consecutive epochs.

5 Experiments and Results

5.1 Baselines

Pre-trained $BERT_{BASE}$ model is the base model f_θ for all the models in this paper. We consider Prototypical Networks by (Fritzler et al., 2019) without fine-tuning as a baseline. Another baseline is the 5-shot nearest neighbor (NNShot) model, as described in section 4.4.

5.2 ProtoNet vs MAML

We compare the 5-shot performance of our ProtoNet and the MAML model. For MAML, we use an outer loop learning rate (for ϕ) of 0.001 and meta-batch size of 4. In this setting, we train all the layers (L = 12) in the base model. ProtoNet performs slightly better with 46.1% F1-score as compared to MAML with 43.3% F1-score as illustrated in Table 2. We find that ProtNet trained with BERT-base model performs best. If we do not consider O-tags for both training and validation episodes, we obtain an F1-score of 73.1% for Prototypical Network and 64% for MAML.

5.3 Effect of distance metric

We analyze the performance of our model using the hyperbolic distance metric used in Wohlwend et al. (2019) and the Euclidean distance metric. The experimental results are shown in Table 3 where we varied the trainable layers in the BERT model.

We find that the hyperbolic distance metric performs better when the base model is completely frozen. Euclidean metric performs better when layers of the base model are also trained. We assume

Model	F1-Scores
Prototypical Network	36.4
NNShot	45.3
MaML	43.3
ProtoNet (Ours)	**46.1**

Table 2: Five-shot episodic evaluation on OntoNotes dataset.

Metric	Output Dim (d)	Frozen Layers (L)						
		0	**2**	**4**	**6**	**8**	**10**	**12**
Hyperbolic	192	**63.40**	64.13	64.57	65.51	66.82	67.66	66.56
Euclidean		60.59	**64.69**	**68.16**	**69.63**	**70.75**	**70.80**	**70.03**
Hyperbolic	512	**67.90**	64.65	68.08	68.20	69.69	71.40	68.86
Euclidean		65.49	**69.29**	**71.46**	**71.41**	**72.65**	**72.43**	**72.72**
Hyperbolic	2048	**70.47**	68.23	73.00	73.80	74.58	-	**74.91**
Euclidean		68.92	**72.66**	**74.69**	**75.31**	**75.45**	-	74.90
Hyperbolic	4096	64.52	70.68	74.40	75.74	**76.59**	-	-
Euclidean		**66.31**	**73.47**	**75.45**	**75.96**	76.49	-	-

Table 3: Analysis on the *output dimension* and the *distance metric* with *trainable BERT layers* for epi. **w/o** O-tag

this is because BERT model is heavily pre-trained to work well with euclidean distance metric.

5.4 Effect of output dimensions (d)

We use our ProtoNet model for this analysis. A larger output dimension d for the linear layer in the base model provides a higher degree of freedom of entity representations. A series of experiments were performed to understand the correlation between the model performance and projecting the entity representations to a higher dimensional space. These experimental results are shown in Table 3 where we observe that using higher output dimension is better for ProtoNet.

5.5 Effect of trainable layers (L)

We explore the effects of the number of trainable layers in BERT for few-shot NER setting. In our experiments, we use the OntoNotes data to fine-tune the BERT model with gradual unfreezing of the layers. In figure 2 we see that the models with $L > 6$ are overfitting the training process while the validation performance increases only marginally (+2 F1 points compared to $L = 4$). The training process for $L < 6$ shows no overfitting and gives comparable results on validation data.

5.6 Effect of domain-transfer

Most of the existing work employs a supervised approach using labeled data for both training and testing. This causes the model to under-perform for domains with no NER data. We perform experiments to study the domain transfer capabilities of our few-shot NER model. For these experiments, we train our model on the Ontonotes dataset and test the performance of the model on i2b2 and Wnut datasets. We chose Ontonotes for training as it represents a more general domain when compared to other datasets in consideration.

Model	I2B2'14	WNUT'17
Prototypical Network	27.6 ± 2.8	22.1 ± 3.1
NNShot	45.4 ± 3.2	26.7 ± 4.0
StructShot	46.1 ± 3.2	27.9 ± 3.2
Ours($d = 192$, $L = 0$)	67.26	23.19
Ours($d = 192$, $L = 12$)	**80.31**	**32.23**
Ours($d = 2048$, $L = 0$)	73.75	25.13
Ours($d = 2048$, $L = 12$)	**81.92**	**33.83**

Table 4: Comparing episodic performance (F1 scores) on domain transfer with our model, *Prtotypical Network, StructShot (Yang and Katiyar, 2020), NNShot*

We see an increment in the performance of the model uptill four layers, after which performance remains stagnant. Interestingly, this result is consistent with the results of (Tenney et al., 2019) 4-6 layers of BERT-base model has a larger correlation with the NER task. Thus our experiments reinforce the layer-wise importance of NER tasks.

6 Conclusion

In this work, we perform detailed analysis of the NER task in few-shot learning setting in four different dimensions. This thorough analysis investigates effects of trainable layers depth, effect of output dimensions, type of distance metric. Expanding on the prototypical networks as our baseline, ProtoNet (Ours) shows SOTA performance for few-shot learning on OntoNotes. We also show SOTA performance on domain-transfer setting on WNUT and I2B2 datasets.

In future work, an interesting exploration would be to combine the insights from this paper and apply them further to Meta-Learning and Online Continual learning of entities on structured text.

References

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1(Mlm):4171–4186.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Edward Grefenstette, Brandon Amos, Denis Yarats, Phu Mon Htut, Artem Molchanov, Franziska Meier, Douwe Kiela, Kyunghyun Cho, and Soumith Chintala. 2019. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Nithin Holla, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020. Learning to learn to disambiguate: Meta-learning for few-shot word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4517–4533, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works.

J. Snell, Kevin Swersky, and R. Zemel. 2017. Prototypical networks for few-shot learning. *ArXiv*, abs/1703.05175.

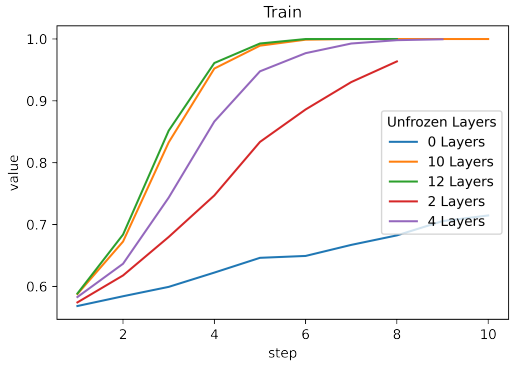
Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Jeremy Wohlwend, Ethan R. Elenberg, Samuel Altschul, Shawn Henry, and Tao Lei. 2019. Metric Learning for Dynamic Text Classification. pages 143–152. Association for Computational Linguistics (ACL).

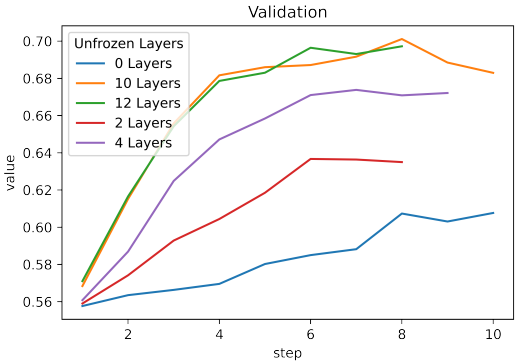
Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

A Appendices

In Figure 2, we show how the training process overfits when we use $L > 6$ layers for episodic few-shot training.



(a) Training vs Epoch



(b) Validation vs Epoch

Figure 2: Effect of freezing BERT layers

B Setup comparison with Yang and Katiyar (2020)

All our experiments are with **6-way 5-shot** setup and we try to compare episodic evaluation.

Our **OntoNotes tag-extension** and **WNUT domain-transfer** set ups are **matching** exactly with Yang and Katiyar (2020) because the test sets have exactly 6 classes.

However, domain transfer setting for i2b2 is not matching as there are 24 entity classes in **I2B2'14**

C Linear layer

The **linear layer** after BERT is actually degrading performance. Upon removal of the linear layer, the performance is shown in Fig 5

Model	F1-Scores
Prototypical Network	36.4
NNShot	45.3
MaML	56.2
ProtoNet (Ours)	60.9

Table 5: Five-shot episodic evaluation on OntoNotes dataset with O-Tag in episodes.