Hierarchical Attention Prototypical Networks for Few-Shot Text Classification

Shengli Sun[1*] **Qingfeng Sun**[1*] **Kevin Zhou**[2 †] **Tengchao Lv**[1]
[1]Peking University [2]Microsoft
slsun@ss.pku.edu.cn {sunqingfeng, lvtengchao}@pku.edu.cn
kezhou@microsoft.com

Abstract

Most of the current effective methods for text classification task are based on large-scale labeled data and a great number of parameters, but when the supervised training data are few and difficult to be collected, these models are not available. In this paper, we propose a hierarchical attention prototypical networks (HAPN) for few-shot text classification. We design the feature level, word level, and instance level multi cross attention for our model to enhance the expressive ability of semantic space. We verify the effectiveness of our model on two standard benchmark few-shot text classification datasets – FewRel and CSID, and achieve the state-of-the-art performance. The visualization of hierarchical attention layers illustrates that our model can capture more important features, words, and instances separately. In addition, our attention mechanism increases support set augmentability and accelerates convergence speed in the training stage.

1 Introduction

The dominant text classification models in deep learning (Kim, 2014; Zhang et al., 2015a; Yang et al., 2016; Wang et al., 2018) require a considerable amount of labeled data to learn a large number of parameters. However, such methods may have difficulty in learning the semantic space in the case that only few data are available. Few-shot learning has became an effective approach to solve this challenge, it can train a neural network with a few parameters using few data but achieve good performance. A typical example of this approach is prototypical networks (Snell et al., 2017), which averages the vector of few support instances as the class prototype and computes distance between target query and each prototype, then classify the query to the nearest prototype's class. However, prototypical networks is rough and does not consider the adverse effects of various noises in the data, which weakens the discrimination and expressiveness of the prototype.

In this paper, we propose a hierarchical attention prototypical networks for few-shot text classification by using attention mechanism in three levels. For feature level attention, we use convolutional neural networks to get the feature scores which is different for various classes. For word level attention, we adopt an attention mechanism to learn the importance of each word hidden state in an instance. For instance level multi cross attention, with the help of multi cross attention between support set and target query, we can determine the importance of different instances in the same class and enable the model to get a more discriminative prototype of each class.

In the actual scenario, we apply HAPN on intention detection of our open domain chatbots with different character. If we create a chatbot for old people, the user intentions will focus on children, health or expectation, so we can define specific intentions and supply related responses. And because of only few data are needed, we can expand the number of classes quickly. The model helps chatbot to identify user intentions precisely, makes the dialogue process smoother, more knowledgeable and more controllable.

There are three main parts of our contribution: first of all, we propose a hierarchical attention prototypical networks for few-shot text classification, then we achieve state-of-the-art performance on FewRel and CSID datasets, and the experiments prove our model is faster and more extensible.

2 Related Works

2.1 Text Classification

Text Classification is an important task in Natural Language Processing, and many models are

[*] They contribute equally to this work.
[†] Corresponding author.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 476–485, Hong Kong, China, November 3–7, 2019.

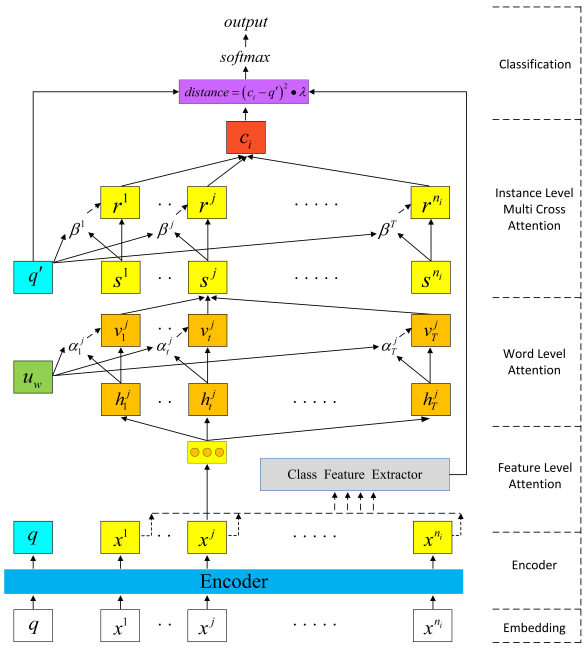


Figure 1: Hierarchical Attention Prototypical Networks architecture

proposed to solve it. The traditional methods mainly focus on feature engineerings such as bag-of-words or n-grams (Wang and Manning, 2012) or SVMs (Tang et al., 2015). The neural network based methods like Kim (2014) applies convolutional neural networks for sentence classification. Then, Johnson and Zhang (2015) use a one-hot word order CNN, and Zhang et al. (2015b) apply a character level CNN. C-LSTM (Zhou et al., 2015) combines CNN and RNN for sentence representation and text classification. Yang et al. (2016) explore the hierarchical structure of documents classification, they use a GRU-based attention to build representations of sentences and another GRU-based attention to aggregate them into a document representation. But above supervised learning methods require large-scale labeled data and can't classify unseen classes.

2.2 Few-Shot Learning

Few-Shot Learning (FSL) aims to solve classification problems by training a classifier with few instances in each class, and it can apply to unseen classes. The early works aim to use transfer learning approaches, Caruana (1994) and Bengio (2011) adopt the target task from the pre-trained models. Then Koch et al. (2015) explore a method for learning siamese neural networks which employs an unique structure to rank similarity between inputs. Vinyals et al. (2016) use matching networks to map a small labeled support set and an unlabelled example to its label, and obviate the need for fine-tuning to adapt to new class types. Prototypical networks (Snell et al., 2017) learns a metric space in which the model can perform well by computing distance between query and prototype representations of each class and classify the query to the nearest prototype's class. Sung et al. (2018) propose a two-branch relation networks, which learns to compare query against few-shot labeled sample support data. Dual TriNet structure (Chen et al., 2018) can efficiently and directly augment multi-layer visual features to boost the few-shot classification.But all of the above works mainly concentrate on computer vision field, the research and applications in NLP field are extremely limited. Recently, Yu et al. (2018) propose an adaptive metric learning approach that automatically determines the best weighted combination from a set of metrics obtained from meta-training tasks for a newly seen few-shot task such as intention classification, Han et al. (2018) present a relation classification dataset – FewRel, and adapt most recent state-of-the-art few-shot learning methods for it, Gao et al. (2019) propose a hybrid attention-based prototypical networks for noisy few-shot relation classification. However, these methods do not consider mining semantic information or reducing the impact of noise more precisely. And in most of the realistic settings, we may increase the number of instances gradually, so model capacity needs more attention.

3 Task Definition

In few-shot text classification task, our goal is to learn a function : $G(D, S, x) \rightarrow y$. D is the labeled data, we divide D into three parts: D_{train}, $D_{validation}$, and D_{test}, and each part has specific label space. We use D_{train} to optimize parameters, $D_{validation}$ to select best hyper parameters, and D_{test} to evaluate the model.

The "episode" training strategy that Vinyals et al. (2016) proposed has proved to be effective. For each training episode, we first sample a label set $\mathcal{L}$ from D_{train}, then use $\mathcal{L}$ to sample the support set S and the query set Q, finally, we feed S and Q to the model and minimize the loss. If $\mathcal{L}$ includes N different classes and each class of S contains K instances, we call the target problem N-way K-shot learning. For this paper, we consider N = 5 or 10, and K = 5 or 10.

For exactly, in an episode, we are given a support set S

$$\begin{aligned} S = &\{(x_1^1, l_1), (x_1^2, l_1), \ldots, (x_1^{n_1}, l_1), \\ &\cdots, \\ &(x_m^1, l_m), (x_m^2, l_m), \ldots, (x_m^{n_m}, l_m)\}, \\ &l_1, l_2, \cdots, l_m \in \mathcal{L} \end{aligned} \tag{1}$$

consists of n_i text instances for each class $l_i \in \mathcal{L}$, x_i^j means it is the j support instance belonging to calss l_i, and instance x_i^j includes $T_{i,j}$ words $\{w_1, w_2, \ldots, w_{T_{i,j}}\}$.

Then x is an unlabeled instance of query set Q to classify, and $y \in \mathcal{L}$ is the output label followed by the prediction of G.

4 Method

4.1 Model Overview

The overall architecture of the Hierarchical Attention Prototypical Networks is shown in Figure 1. We introduce different components in the following subsections:

Instance Encoder Each instance in support set or query set will be first represented to a input vector by transforming each word into embeddings. Considering the lightweight and speed of the model, we achieve this part with one layer convolutional neural networks (CNN). For ease of comparison, its details are the same as Han et al. (2018) proposed.

Hierarchical Attention In order to get more important information from rare data, we adopt a hierarchical attention mechanism. Feature level attention enhances or reduces the importance of different feature in each class, word level attention highlight the important words for meaning of the instance, and instance level multi cross attention can extract the important support instances for different query instances, these three attention mechanisms work together to improve the classification performance of our model.

Prototypical Networks Prototypical networks compute a prototype vector as the representation of each class, and this vector is the mean vector of the embedded support instances belonging to its class. We compare the distance between all prototype vectors and a target query vector, then classify this query to the nearest one.

4.2 Instance Encoder

The instance encoder part consists of two layers: embedding layer and instance encoding layer.

4.2.1 Embedding Layer

Given an instance $x = \{w_t, w_2, \ldots, w_T\}$ with T words. We use an embedding matrix W_E,$\boldsymbol{w}_t = W_E w_t$ to embed each word to a vector

$$\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_T\}, \boldsymbol{w}_t \in \mathbb{R}^d \tag{2}$$

where d is the word embedding dimension.

4.2.2 Encoding Layer

Following we apply a convolutional neural network Zeng et al. (2014) as encoding layer to get the hidden annotations of each word by a convolution kernel with the window size m

$$\boldsymbol{h}_t = \text{CNN}(\boldsymbol{w}_{t-\frac{m-1}{2}}, \ldots, \boldsymbol{w}_{t-\frac{m+1}{2}}) \tag{3}$$

Especially, if the word w_t has a position embedding $\boldsymbol{p}_t$, we should concat $\boldsymbol{w}_t$ and $\boldsymbol{p}_t$

$$\boldsymbol{wp}_t = [\boldsymbol{w}_t \oplus \boldsymbol{p}_t] \tag{4}$$

where $\oplus$ is a concatation, the $\boldsymbol{h}_t$ will be as follow

$$\boldsymbol{h}_t = \text{CNN}(\boldsymbol{wp}_{t-\frac{m-1}{2}}, \ldots, \boldsymbol{wp}_{t-\frac{m+1}{2}}) \tag{5}$$

Then, we aggregate all $\boldsymbol{h}_t$ to get the overall representation of instance x

$$\boldsymbol{x} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_t\} \tag{6}$$

Finally, we define those two layers as a comprehensive function

$$\boldsymbol{x} = g_\theta(x) \tag{7}$$

θ in this function are the networks parameters to be learned.

4.3 Prototypical Networks

The prototypical networks (Snell et al., 2017) has achieved excellent performance in few-shot image classification and few-shot text classification (Han et al., 2018; Gao et al., 2019) tasks respectively, so our model is based on prototypical networks and aims to get promotion.

The fundamental idea of prototypical networks is simple but efficient: we can use a prototype vector $\boldsymbol{c}_i$ as the representative feature of class l_i, each prototype vector can be calculated by averaging all the embedded instances in its support set

$$\boldsymbol{c}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} g_\theta(x_i^j) \tag{8}$$

Then the probability distribution over the classes in $\mathcal{L}$ can be produced by a softmax function over distances between all prototypes vector and the target query q

$$p_\theta(y = l_i|q) = \frac{\exp(-d(g_\theta(q), \boldsymbol{c}_i)}{\Sigma_{l=1}^{|\mathcal{L}|} \exp(-d(g_\theta(q), \boldsymbol{c}_l)} \tag{9}$$

As Snell et al. (2017) mentioned, squared Euclidean distance is a reasonable choice, however, we will introduce a more effective method in section 4.4.1, which combines squared Euclidean distance with class feature scores, and achieves definite improvement.

4.4 Hierarchical Attention

We focus on sentence-level text classification in this work. The proposed model gets a feature scores vector and transfers the support set of each class into a vector representation, on which we build a classifier to perform few-shot text classification.

4.4.1 Feature Level Attention

Obviously, the same dimension belonging to different classes has different importance when we calculate the euclidean distance. In other words, some feature dimensions are more discriminative for distinguishing specific class in the feature level space, and other features are confusing and useless at the same time.

So we apply a CNN-based feature attention mechanism similar to Gao et al. (2019) proposed as a class feature extractor. It depends on all the instances in the support set of each class and will dynamiclly change with different classes.

Given a support set $\boldsymbol{S}_i \in \mathbb{R}^{n_i \times T \times d}$ of class l_i as the output of above instance encoder part

$$\boldsymbol{S}_i = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{n_i}\} \tag{10}$$

we apply a max pooling layer over each instance in $\boldsymbol{S}_i$ to get a new feature map $\boldsymbol{S}_{\boldsymbol{c}i} \in \mathbb{R}^{n_i \times d}$. Then we use three convolution layers to obtain $\boldsymbol{\lambda}_i \in \mathbb{R}^d$, which is the scores vector of class l_i. The specific structure of above class feature extractor is shown in Table 1.

layer name	kernel size	stride	output size
pool	$T \times 1$	1×1	$K \times d \times 1$
conv_1	$K \times 1$	1×1	$K \times d \times 32$
	ReLU		
conv_2	$K \times 1$	1×1	$K \times d \times 64$
	ReLU		
conv_3	$K \times 1$	$K \times 1$	$1 \times d \times 1$
	ReLU		

Table 1: Class feature extractor architecture

So we get a new distance calculation method as follow

$$d(\boldsymbol{c}_i, \boldsymbol{q}^{'}) = (\boldsymbol{c}_i - \boldsymbol{q}^{'})^2 \cdot \boldsymbol{\lambda}_i \tag{11}$$

where $\boldsymbol{q}^{'}$ is the query vector passed through the word level attention mechanism which will be introduced in the next subsection.

4.4.2 Word Level Attention

The importance of different words to the meanings of an instance is unequal, thus it is worth pointing out which words are useful and which words are useless. Therefore, we apply an attention mechanism (Yang et al., 2016) to get those important words and assemble them to compose a more informative instance vector $\boldsymbol{s}^j$, and the definitions are as follows

$$\boldsymbol{u}_t^j = \tanh(\boldsymbol{W}_w \boldsymbol{h}_t^j + \boldsymbol{b}_w) \tag{12}$$

$$v_t^j = \boldsymbol{u}_t^{j\top} \boldsymbol{u}_w \tag{13}$$

$$\alpha_t^j = \frac{\exp(v_t^j)}{\Sigma_t \exp(v_t^j)} \tag{14}$$

$$\boldsymbol{s}^j = \sum_t \alpha_t^j \boldsymbol{h}_t^j \tag{15}$$

where $\boldsymbol{h}_t^j$ is the t hidden word embedding of instance $\boldsymbol{x}^j$, it was encoded through the instance encoder, and has the same hidden size with $\boldsymbol{x}^j$.

Firstly, the $\boldsymbol{W}_w$ and $\boldsymbol{b}_w$ followed by activation function $\tanh$ make up a MLP layer to transform $\boldsymbol{h}_t^j$ to the new hidden representation $\boldsymbol{u}_t^j$. Immediately, we apply a dot product operation between $\boldsymbol{u}_t^j$ and a word level weight vector $\boldsymbol{u}_w$ to compute similarity v_t^j as the importance weight of $\boldsymbol{u}_t^j$. Then we use a softmax function to normalize v_t^j to α_t^j. Finally, we calculate the instance level vector $\boldsymbol{s}^j$ through the weighted sum of α_t^j and $\boldsymbol{h}_t^j$. As memory networks (Sukhbaatar et al., 2015) proposed, $\boldsymbol{u}_w$ can help us to select the important words in each instance, it will be randomly initialized at the beginning of the training stage, and be optimized together with the networks parameters θ.

4.4.3 Instance Level Multi Cross Attention

The previous prototypical networks use the mean vector of support instances as the class prototype. Because of the diversity and lack of the support instances, the gap between each support vector and prototype maybe wide, meanwhile, different query instances can be expressed in several ways, so not every instance in a support set contributes equally to the class prototype when they face a target query instance. To highlight the importance of support instances which are useful clues to classify a query instance correctly, we propose a multi cross attention mechanism.

Given a support set $\boldsymbol{S}_i^{'} \in \mathbb{R}^{n_i \times d}$ for class l_i and a query vector $\boldsymbol{q}^{'} \in \mathbb{R}^d$, they are all encoded through the instance encoder and word level attention. We consider each support vector $\boldsymbol{s}_i^j$ in $\boldsymbol{S}_i^{'}$ has its own weight β_i^j to query $\boldsymbol{q}^{'}$. So the formula (8) will be rewritten as follow

$$\boldsymbol{c}_i = \sum_{j=1}^{n_i} \beta_i^j \boldsymbol{s}_i^j \tag{16}$$

where we define $\boldsymbol{r}_i^j = \beta_i^j \boldsymbol{s}_i^j$ as the weighted prototype vector and the definitions of β_i^j are as follows

$$\beta_i^j = \frac{\exp(\gamma_i^j)}{\Sigma_{j=1}^{n_i} \exp(\gamma_i^j)} \tag{17}$$

$$\gamma_i^j = \text{sum}\{\sigma(f_\varphi(mca))\} \tag{18}$$

$$mca = [\boldsymbol{s}_{i\phi}^j \oplus \boldsymbol{q}_\phi^{'} \oplus \boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2] \tag{19}$$

$$\boldsymbol{\tau}_1 = |\boldsymbol{s}_{i\phi}^j - \boldsymbol{q}_\phi^{'}|, \boldsymbol{\tau}_2 = \boldsymbol{s}_{i\phi}^j \odot \boldsymbol{q}_\phi^{'} \tag{20}$$

$$\boldsymbol{s}_{i\phi}^j = f_\phi(\boldsymbol{s}_i^j), \boldsymbol{q}_\phi^{'} = f_\phi(\boldsymbol{q}^{'}) \tag{21}$$

where f_ϕ is a linear layer, $|\cdot|$ is element-wise absolute value and $\odot$ is element-wise product, we use these two operation to get the difference information $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$ between $\boldsymbol{s}_i^j$ and $\boldsymbol{q}^{'}$, then concatenate them all as the multi cross attention information mca, then $f_\varphi(\cdot)$ is a linear layer, $\sigma(\cdot)$ is a $tanh$ activation function, sum$\{\cdot\}$ means a sum operation of all elements in the vector. Finally, γ_i^j is the weight of j instance in support set $\boldsymbol{s}_i$, and we use a softmax function to nomalize it to β_i^j.

Through the multi cross attention mechanism, the prototype can pay more attention to those query-related support instances and improve the capacity of support set.

5 Experiments

In this section, we will introduce the experiment results of our model. Firstly, we evaluate our model on FewRel dataset and CSID dataset, and achieve state-of-the-art results, our model outperforms the best baselines models by 1.11% and 1.64% respectively on 10 way 5 shot setting. Then we will show how our model works by case study and visualization of attention layers. We further demonstrate that the hierarchical attention increases the augmentability of support set and the convergence speed of the model.

5.1 Datasets

FewRel Few-Shot Relation Classification (Han et al., 2018) is a new large-scale supervised dataset[1]. It consists of 70000 instances on 100 relations derived from Wikipedia, and each relation includes 700 instances. It also marks the head and tail entities in each instance, and the average number of tokens is 24.99. FewRel has 64 relations for training, 16 relations for validation, and 20 relations for test separately.

CSID Character Studio Intention Detection is a dataset extracted from a real-world open domain chatbot. In character studio platform, this chatbot should transform its character style sometime so it can adapt to different user group and environment, thus dialog query intention detection turns into an important task. CSID consists of 24596 instances for 128 intentions, and each intention includes 30 to 260 instances, the average number of tokens in each instance is 11.52. We use 80, 18 and 30 intentions for training, validation, and test respectively.

[1] https://github.com/thunlp/FewRel

Model	5 Way 5 Shot	5 Way 10 Shot	10 Way 5 Shot	10 Way 10 Shot
Finetune	69.43 ± 0.30	71.56 ± 0.33	58.31 ± 0.26	62.12 ± 0.38
kNN	71.94 ± 0.33	73.20 ± 0.35	61.69 ± 0.36	66.49 ± 0.42
MetaN	79.46 ± 0.35	83.63 ± 0.38	70.49 ± 0.52	72.84 ± 0.69
GNN	80.42 ± 0.56	83.50 ± 0.63	63.08 ± 0.70	64.81 ± 0.85
SNAIL	78.64 ± 0.19	81.22 ± 0.23	69.46 ± 0.25	71.29 ± 0.27
Proto	83.79 ± 0.12	86.05 ± 0.10	75.25 ± 0.14	77.14 ± 0.19
PHATT	86.96 ± 0.05	87.56 ± 0.08	78.62 ± 0.11	80.94 ± 0.14
HAPN-FA	86.53 ± 0.07	87.05 ± 0.07	78.23 ± 0.09	80.73 ± 0.12
HAPN-WA	87.91 ± 0.09	87.83 ± 0.12	79.31 ± 0.10	81.06 ± 0.17
HAPN-IMCA	88.07 ± 0.09	88.96 ± 0.11	80.02 ± 0.12	81.87 ± 0.15
HAPN	**88.45 ± 0.06**	**89.72 ± 0.08**	**80.26 ± 0.11**	**82.68 ± 0.13**

Table 2: Accuracies (%) of different models on the CSID dataset on four different settings.

5.2 Baselines

Firstly, we compare our model with several traditional models such as Finetune and kNN, Then we compare our model with five state-of-the-art few-shot learning models based on neural networks, they are MetaN (Munkhdalai and Yu, 2017), GNN (Garcia and Bruna, 2018), SNAIL (Mishra et al., 2018), Proto (Snell et al., 2017) and PHATT (Gao et al., 2019) respectively.

5.3 Implementation details

We compare our models with seven baselines, and the implementation details are as follows.

For FewRel dataset, we cite the results reported by Snell et al. (2017) which includes Finetune, kNN, MetaN, GNN, and SNAIL, then we cite the results reported by Gao et al. (2019) which includes Proto and PHATT. For a fair comparison, in our model, we use the same word embeddings and hyperparameters of instance encoder as PHATT proposed. In detail, we use the Glove (Pennington et al., 2014) consisting of 6B tokens and 400K vocabulary as our initialized word representation, and each word has a 50 dimensions vector. In addition, the position embedding dimension of a word is 10, the max length of each instance is 40. Finally, we evaluate all models on 5 way 5 shot and 10 way 5 shot settings.

For CSID dataset, we implement all above seven baseline models and our models. we use the Baidu Encyclopedia (Li et al., 2018) as our initialized word representation, it includes 745M tokens and 5422K vocabulary, and each word has a 300d dimensions vector, the max length of each instance is 20. Finally, we evaluate all models on 5 way 5 shot, 5 way 10 shot, 10 way 5 shot and 10 way 10 shot settings.

For the Finetune and kNN baselines, they learn the parameters on the support set with the CNN encoder. For the neural networks based baselines, we use the same hyper parameters as Han et al. (2018) proposed.

For our hierarchical attention prototypical networks, the window size of the CNN instance encoder is 3, the dimension of the hidden layer is 230, the learning rate is 0.1, the learning rate decay step is 3000 and the decay rate is 0.1. In addition, we train our model 12000 episodes and each episode consists of 20 classes.

In order to study the effects of different components, we refer to our models as HAPN-{FA,WA, IMCA}, FA indicates feature level attention, WA indicates word level attention and IMCA indicates instance level multi cross attention.

5.4 Results and analysis

The experimental accuracies on CSID and FewRel are shown in Tabel 2 and Table 4 respectively. In this subsection, we will show the effects of hierarchical attention and support set augmentability of three Proto-based models and the convergence speed comparison.

5.4.1 Effects of hierarchical attention

Benefit from hierarchical attention, our model achieves excellent performance.

The case study of word level attention and instance level multi cross attention are shown in Table 3, this is a 2 way 3 shot task on FewRel dataset. The query instance is an instance of “mother” class in fact, and our model should classify it into “mother” class or “child” class. It is a difficult

Class	Word Attention	IMCAS
	Support Set	
(1) mother	**Cherie Gil** is the daughter of Filipino actors Eddie Mesa and **Rosemarie Gil**, and sister of fellow actors, Michael de Mesa and the late Mark Gil.	
	When they reached adulthood, Pelias and **Neleus** found their mother **Tyro** and then killed her stepmother, Sidero, for having mistreated her.	
	It was here that the Queen Consort **Jetsun Pema** gave birth to a son on 5 February 2016, **Jigme Namgyel Wangchuck**.	
(2) child	In 1421 **Mehmed** died and his son **Murad II** refused to honour his father's obligations to the Byzantines.	
	Henry Norreys was a lifelong friend of Queen Elizabeth and was the father of six sons, who included Sir **John Norreys**, a famous English soldier.	
	Jim Henson and his son **Brian** were impressed enough with Barron's style to offer him a job directing the pilot episode of "The Storyteller".	
	Query	
(1) or (2)	From 1922 to 1963, **Princess Dagmar of Demark**, the daughter of Frederick VIII of Denmark and **Louise of Sweden**, lived on Kongestlund.	

Table 3: Visualization of word level and instance level multi cross attention scores (IMCAS) for 2 way 3 shot setting, the bold words are head entities and tail entities.

Model	5 Way 5 Shot	10 Way 5 Shot
Finetune*	68.66 ± 0.41	55.04 ± 0.31
kNN*	68.77 ± 0.41	55.87 ± 0.31
MetaN*	80.57 ± 0.48	69.23 ± 0.52
GNN*	81.28 ± 0.62	64.02 ± 0.77
SNAIL*	79.40 ± 0.22	68.33 ± 0.25
Proto◇	89.05 ± 0.09	81.46 ± 0.13
PHATT◇	90.12 ± 0.04	83.05 ± 0.05
HAPN-FA	89.79 ± 0.13	82.47 ± 0.20
HAPN-WA	90.86 ± 0.12	83.79 ± 0.19
HAPN-IMCA	90.92 ± 0.11	84.07 ± 0.19
HAPN	**91.02 ± 0.11**	**84.16 ± 0.18**

Table 4: Accuracies (%) for 5 way 5 shot and 10 way 5 shot settings on FewRel test set. * reported by Han et al. (2018) and ◇ reported by Gao et al. (2019).

task because of there are many similarities between the expressions of two classes. With the help of word level attention, we highlight the importance of the word "daughter", which appears in the query instance and the first support instance of class "mother" at the same time, then this support instance get the highest attention score and contributes more to the prototype vector of "mother" class, finally our model can classify the query instance into the correct class in this confusing task.

As shown in Figure 2, by using the feature level attention, we also get the feature attention scores of "mother" class and "child" class respectively. The features with high scores have deep color, and the features with low scores have light color. Obviously, different classes may have different feature score vector, in other words, the same feature of different classes have different importance. So our feature level attention can highlight importance of the useful features and weaken the importance of the noise features, then the distance between the prototype vector and the query vector will measure the difference between them more efficiently.

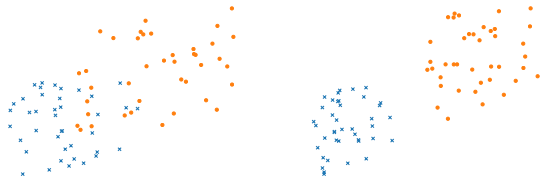
(a) Feature attention scores of "mother" class

(b) Feature attention scores of "child" class

Figure 2: Feature attention scores of different classes

We treat the final prototype embedding vector as the features of each instance, then we can get the distribution of features by principal pomponent analysis in feature space as shown in Figure 3. As we can see, the instances without hierarchical attention are more distributed and may cross with each other, but the instances with hierarchical attention are more centralized and discriminative, which proves that our model learns a better semantic space, which helps to distinguish confus-

ing data..



(a) Instances without hierarchical attention

(b) Instances with hierarchical attention

Figure 3: Instances distribution of embedding vector without hierarchical attention (a) and with hierarchical attention (b). The left blue points marked × are instances of "mother" class and the right orange points marked • are instances of "child" class.

5.4.2 Augmentability of support set

More support instances can contribute more useful information to the prototype vector, meanwhile, more noise will be added in.

In this section, we define the support set augmentability (SSA) as the additive value of accuracy when we increase the same number of the support set for different models. So we compare our model's SSA with other models such as Proto and PHATT on the 10 way FewRel task, and the shot number ranges from 5 to 25.

By using the hierarchical attention, our model obtaines a strong robustness and can pay more attention to the important information of support set and reduce those negative effects of noisy data, thus as shown in Figure 4, the support set augmentability of our model is larger than other models. Benefit from the above advantages, we can deploy our model in the cold start stage, and gradually accumulate labeled support data in practical applications, then improve the performance of the model day by day, and thus improve the utilization rate of few data in realistic settings.

5.4.3 Convergence speed comparison

At the training stage, we also compare the convergence speed between Proto, PHATT, and HAPN on the 10 way 5 shot and 10 way 15 shot FewRel task. As shown in Figure 5, our model can be optimized more quickly than the other models. From 10 way 5 shot task to 10 way 15 shot settings, the Proto model takes almost twice time to achieve 70% accuracy on validation set, in other words, the convergence speed will decrease sharply when we increase the number of support instances, but this problem can be effectively alleviated when we use hierarchical attention mechanism.

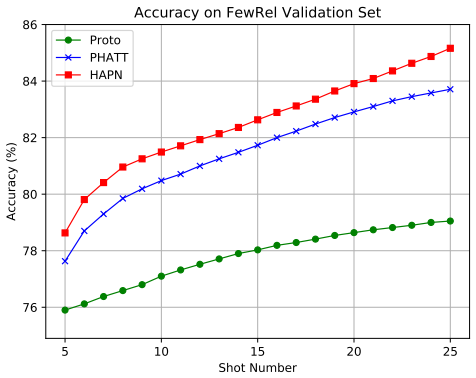


Figure 4: Support set augmentability of Proto, PHATT and HAPN on FewRel validation set.

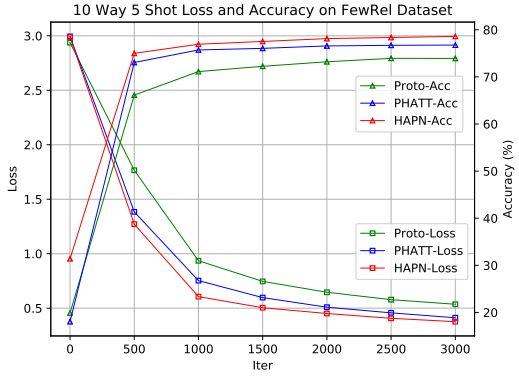


Figure 5: Training Proto, PHATT and HAPN on FewRel dataset. Lines marked □ denote loss on the training set and lines marked △ denote accuracy on the validation set.

6 Conclusion

Previous few-shot learning models for text classification roughly apply text representations or neglect the noisy information. We propose to do hierarchical attention prototypical networks consisting of feature level, word level and instance level multi cross attention, which highlight the important information of few data and learn a more discriminative prototype representation. In the experiments, our model achieves the state-of-the-art performance on FewRel and CSID datasets. HAPN not only increases support set augmentability but also accelerates convergence speed in the training stage.

In the future, we will contribute new text dataset to few-shot learning, explore better feature extrac-

tor networks and do some industrial application.

Acknowledgements

We would like to thank Sawyer Zeng and Yue Liu for providing valuable hardware support and useful advice, and thank Xuexiang Xu and Yang Bai for helping us test online FewRel dataset. This work is also supported by the National Key Research and Development Program of China (No. 2018YFB1402902 and No. 2018YFB1403002) and the Natural Science Foundation of Jiangsu Province (No. BK20151132).

References

Yoshua Bengio. 2011. Deep learning of representations for unsupervised and transfer learning. In *Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011*, pages 17–36.

Rich Caruana. 1994. Learning many related tasks at the same time with backpropagation. In *Advances in Neural Information Processing Systems 7, [NIPS Conference, Denver, Colorado, USA, 1994]*, pages 657–664.

Zitian Chen, Yanwei Fu, Yinda Zhang, Yu-Gang Jiang, Xiangyang Xue, and Leonid Sigal. 2018. Semantic feature augmentation in few-shot learning. volume abs/1804.05298.

Tianyu Gao, Zhiyuan Liu Xu Han, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *Proceedings of the Association for the Advancement of Artificial Intelligence*.

Victor Garcia and Joan Bruna. 2018. Few-shot learning with graph neural networks. In *Proceedings of ICLR*.

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, Brussels, Belgium, October 31 - November 4, 2018*, pages 4803–4809.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 103–112.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv:1408.5822.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning workshop*.

Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical reasoning on chinese morphological and semantic relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 138–143. Association for Computational Linguistics.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A simple neural attentive meta-learner. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2554–2563.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4080–4090.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. Yara parser: A fast and accurate dependency parser. *End-to-end memory networks*, arXiv:1503.08895.

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 *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1199–1208.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1422–1432.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual*

Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, pages 3630–3638.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2321–2331.

Sida I. Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*, pages 90–94.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1480–1489.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1206–1215.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2335–2344.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. A C-LSTM neural network for text classification. volume abs/1511.08630.