A Meta Graph Learning Algorithm

We demonstrate the Meta Graph Learning (MGL) method in Algorithm 1. The whole process consists of three stages: (1) Meta-train on the source languages; (2) Meta-test on the target language; (3) Evaluate on the target language.

B Experimental Environment

Our MGL model and baseline methods are all using Tensorflow 2.0.0 based on CUDA 10.0, single V100 GPU, and are tested on Linux, Python 3.7.4 from Anaconda 4.7.12.

C Baselines

We provide the available open source code for the baseline methods we compare with, including: **CL-SCL**[1] (Prettenhofer and Stein, 2010); MUSE[2] (Conneau et al., 2018) **BWE**; MAN-MoE (Chen et al., 2019)[3]; **Reptile**[4] (Nichol et al., 2018). For other methods, we ask for the source code from authors or directly report the results in their original paper.

D Multilingual Search Relevance Dataset

For detail, we provide the statistics of the large-scale industrial multilingual search relevance dataset (Ahuja et al., 2020) in Table 1.

E Hyper-parameter Sensitivity

The proposed MGL method involves several hyper-parameters. Here, we evaluate how different choices of hyper-parameters affect the performance of MGL. In the following, except for the parameter being tested, all other parameters keep the same. Hyper-parameter sensitivity is done in the multilingual Amazon review dataset.

E.1 The number of neighborhoods

We investigate the influence of the number of the neighborhood in the K-nearest meta graph for the proposed MGL model. We use the averaged accuracy among all 9 cross-lingual transfer experiments on the multilingual Amazon review dataset and test the performance with $K = 1, 5, 10, 15, 20$. The results are shown in Figure 1. Increasing K from 1, 5 to 10 shows continuous improvements. When K further increases, the performance stops to improve and tends to be stable. This demonstrates that increasing the number of the neighborhoods in an appropriate interval, i.e., $[1, 10]$, is beneficial for transferring more useful knowledge to improve the learning in the low-resource target language. While further incorporating more neighborhoods cannot enhance the transfer effectiveness since it may bring in some useless heterogeneous information for propagation.

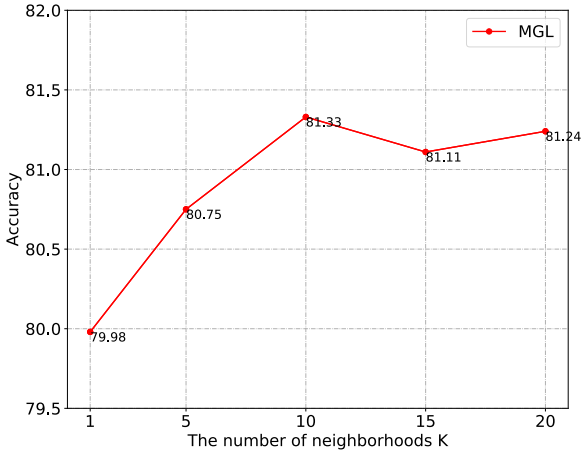

Figure 1: The results w.r.t. the different number of the neighborhoods K on the Amazon review dataset.

[1] https://github.com/pprett/nut

[2] https://github.com/facebookresearch/MUSE

[3] https://github.com/microsoft/Multilingual-Model-Transfer

[4] https://github.com/openai/supervised-reptile

References

Aman Ahuja, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K Reddy. 2020. Language-agnostic representation learning for product search on e-commerce platforms. In *WSDM*, pages 7–15.

Xilun Chen, Ahmed Hassan, Hany Hassan, Wei Wang, and Claire Cardie. 2019. Multi-source cross-lingual model transfer: Learning what to share. In *ACL*, pages 3098–3112.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *ICLR*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *ACL*, pages 1118–1127.

Language	Domain	# Train pairs (+ | −)	# Val pairs (+ | −)	# Test pairs (+ | −)
French (FR)	Search	38,069 (33,499 | 4,570)	4,230 (3,768 | 462)	20,796 (17,930 | 2,866)
Spanish (ES)	Search	53,330 (45,717 | 7,583)	5,923 (5,096 | 827)	23,395 (19,973 | 3,422)
Italian (IT)	Search	106,563 (82,059 | 24,504)	9,088 (9,115 | 2,753)	46,309 (38,908 | 7,401)
English (EN)	Search	282,046 (26,0879 | 21,167)	31,339 (28,980 | 2,359)	48,504 (45,638 | 2,866)
German (DE)	Search	258,522 (241,192 | 17,330)	28,725 (26,798 | 1,927)	64,391 (58,952 | 5,439)

Table 1: The statistics for the multilingual search relevance dataset.

Algorithm 1: Meta Graph Learning (MGL)

Input: Source languages $\{\ell_i^s\}_{i=1}^T$, Target language ℓ^t;
Input: Hyperparameter: learning rate, N, T, K, dim_0, dim_1, $\#train\_episodes$, $\#test\_episodes$, $\#eval\_times$
Output: Performances of the target language

1 Initialize $f_{\boldsymbol{\theta}}$ with the mPLM, e.g., mBERT;
2 Randomly initialize $\boldsymbol{\Theta} = \{\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, \mathbf{W}_\sigma, \mathbf{b}_\sigma\}$;
3 Generate source language pairs $\{p_o^s\}_{o=1}^T$ by *leave-one-out* and the target pair p^t;
/* Meta-train on the source languages */
4 **while** *not done* **do**
5 **for** i=*1, 2, ...,* $\#train\_episodes$ **do**
6 **for** *each pair* p_o^s *in* $\{p_o^s\}_{o=1}^T$ **do**
7 Sample a support set $\mathcal{S}$ and a query set $\mathcal{Q}$ from p_o^s ;
8 Feed $\mathcal{S}$ and $\mathcal{Q}$ into the mPLM encoder $f_{\boldsymbol{\theta}}$ and get the node embedding matrix $\mathbf{H}$;
9 Compute instance-wise length scale parameter $\boldsymbol{\sigma}_j$ by Eq. (1);
10 Compute adjacency matrix $\mathbf{A}$ by Eq. (2) and construct a K-nearest meta graph;
11 Normalize $\mathbf{A}$ and Compute $\mathbf{z}$ by Eq. (4-5);
12 Evaluate $\mathcal{J}$ by Eq. (6);
13 Update $f_{\boldsymbol{\theta}}$ and $\boldsymbol{\Theta}$ by gradient descent;
14 **end**
15 **end**
16 **end**
/* Meta-test on the target language */
17 **while** *not done* **do**
18 **for** i=*1, 2, ...,* $\#test\_episodes$ **do**
19 Sample a support set $\mathcal{S}$ and a query set $\mathcal{Q}$ from p^t;
20 **Repeat** line 8-13;
21 **end**
22 **end**
/* Evaluating on the target language */
23 **for** i = *1,2,...,* $\#eval\_times$ **do**
24 Randomly sample a support set $\mathcal{S}$ from p^t ;
25 **while** *not done* **do**
26 Sample $\mathcal{Q}$ from the testing data of ℓ^t and predict;
27 **end**
28 **end**
29 **Return** Averaged results over $\#eval\_times$;