BGE-中文embedding模型

1. 解决什么问题（动机）
    - 提高中文文本embedding的效果
2. 如何解决
    - 预训练+有监督微调
        - 预训练-无监督微调-有监督微调
    - 数据准备
        - pretrain阶段使用的数据来源于wudao
            - https://www.scidb.cn/en/detail?dataSetId=c6a3fe684227415a9db8e21bac4a15ab
        - 微调数据来源

Table 1: Composition of C-MTP

dataset	C-MTP (unlabeled)	C-MTP (labeled)
source	Wudao, Zhihu, Baike, CSL, XLSUM-Zh, Amazon-Review-Zh, CMRC, etc.	T^2-Ranking, mMARCO-Zh, DuReader, NLI-Zh, etc.
size	$100M$	$838K$

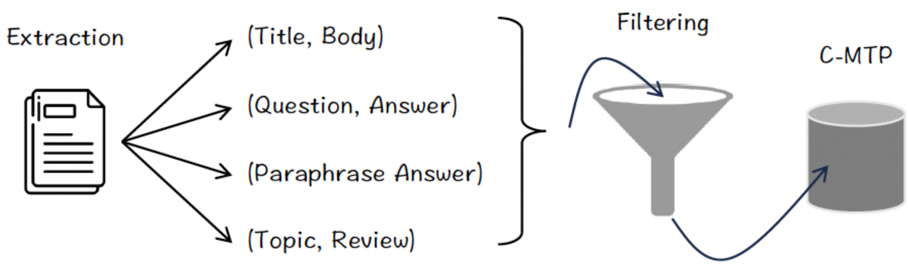


Figure 4: Creation of C-MTP.

- 清洗步骤
    - 去掉非文本、重复、恶意内容
    - 使用Text2Vec-Chinese对文本对进行打分，去掉相似性得分小于0.43的数据
- the text embedding, like retrieval, ranking, similarity comparison, etc. Particularly, the following labeled datasets are included, T^2-Ranking [60], DuReader [20, 42], mMARCO [8], CMedQA-v2[65], multi-cpr[31], NLI-Zh[9], cmnli[62] and ocnli[62]. There are 838,465 paired texts in total, which contains diverse question-answering and paraphrasing patterns. Although it is much smaller than **C-MTP**

- 训练步骤
    - 预训练
        - 预训练，使用Wudao数据集+RetroMAE的方法预训练

$$\min . \sum_{x \in X} -\log \mathrm{Dec}(x|\mathbf{e}_{\tilde{X}}), \ \mathbf{e}_{\tilde{X}} \leftarrow \mathrm{Enc}(\tilde{X}).$$

- RetroMAE:https://arxiv.org/pdf/2205.12035
  - encoder: BERT like encoder with 12 layers and 768 hidden-dimensions
    - a moderate masking ratio (15~30%),
  - decoder:
    - The masking ratio is more aggressive than the one used by the encoder, where 50~70% of the input tokens will be masked

• **Implementation details**. RetroMAE utilizes bi-directional transformers as its encoder, with 12 layers, 768 hidden-dim, and a 30522-token vocabulary (same as BERT base). The decoder is a one-layer transformer. The default masking ratios are 0.3 for encoder and 0.5 for decoder. The model is trained for 8 epochs, with AdamW optimizer, batch-size 32 (per device), learning rate 1e-4. The training is on a machine with 8× Nvidia A100 (40GB) GPUs. The models are implemented with PyTorch 1.8 and HuggingFace transformers 4.16. We adopt the official script [2] from BEIR to prepare the models for their zero-shot evaluation. For super-

- 无监督微调（对比学习）
  - 无监督数据微调，使用对比学习进行微调，负样本采用batch内的其他数据，增大batch size

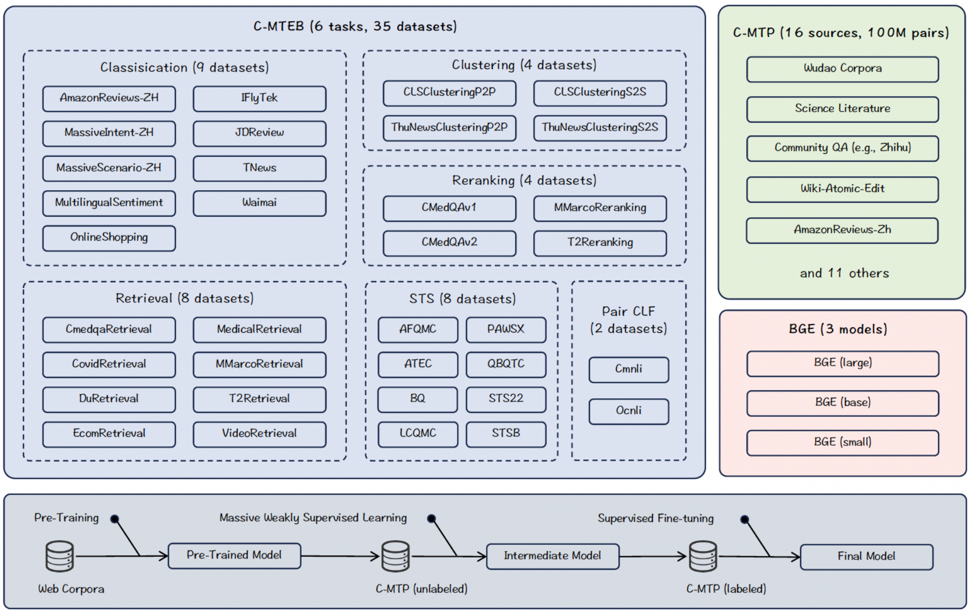
$$\min . \sum_{(p,q)} -\log \frac{e^{\langle \mathbf{e}_p, \mathbf{e}_q \rangle / \tau}}{e^{\langle \mathbf{e}_p, \mathbf{e}_q \rangle / \tau} + \sum_{Q'} e^{\langle \mathbf{e}_p, \mathbf{e}_{q'} \rangle / \tau}}.$$

- 有监督微调
  - 有监督微调，对于负样本采用ANN-style方法找到强负例，指令微调，为不同的任务加入了不同的指令（搜索与该文本相似的embedding）

- **Task-specific fine-tuning**. The embedding model is further fine-tuned with **C-MTP (labeled)**. The labeled datasets are smaller but of higher quality. However, the contained tasks are of different types, whose impacts can be mutually contradicted. In this place, we apply two strategies to mitigate this problem. On one hand, we leverage instruction-based fine-tuning [7, 50], where the input is differentiated to help the model accommodate different tasks. For each text pair (p, q), a task specific instruction I_t is attached to the query side: $q' \leftarrow q + I_t$. The instruction is a verbal prompt, which specifies the nature of the task, e.g., "*search relevant passages for the query*". On the other hand, the negative sampling is updated: in addition to the in-batch negative samples, one hard negative sample q' is mined for each text pair (p, q). The hard negative sample is mined from the task's original corpus, following the ANN-style sampling strategy in [61].

3. 实验结果

- 评估框架



  - STS（Semantic textual similarity)
    - **STS** (Semantic Textual Similarity). The STS [1–5] task is to measure the correlation of two sentences based on their embedding similarity. Following the original setting in Sentence-BERT [46],
    - the Spearman's correlation is computed with the given label, whose result is used as the main metric.
  - Pair CLF
    - **Pair-classification**. This task deals with a pair of input sentences, whose relationship is presented by a binarized label. The relationship is predicted by embedding similarity, where the average precision is used as the main metric.
- 实验结果

- Table 2: Performance of various models on C-MTEB.

model	Dim	Retrieval	STS	Pair CLF	CLF	Re-rank	Cluster	Average
Text2Vec (base)	768	38.79	43.41	67.41	62.19	49.45	37.66	48.59
Text2Vec (large)	1024	41.94	44.97	70.86	60.66	49.16	30.02	48.56
Luotuo (large)	1024	44.40	42.79	66.62	61.0	49.25	44.39	50.12
M3E (base)	768	56.91	50.47	63.99	67.52	59.34	47.68	57.79
M3E (large)	1024	54.75	50.42	64.30	68.20	59.66	**48.88**	57.66
Multi. E5 (base)	768	61.63	46.49	67.07	65.35	54.35	40.68	56.21
Multi. E5 (large)	1024	63.66	48.44	69.89	67.34	56.00	48.23	58.84
OpenAI-Ada-002	1536	52.00	43.35	69.56	64.31	54.28	45.68	53.02
BGE (small)	512	63.07	49.45	70.35	63.64	61.48	45.09	58.28
BGE (base)	768	69.53	54.12	77.50	67.07	64.91	47.63	62.80
BGE (large)	1024	**71.53**	**54.98**	**78.94**	**68.32**	**65.11**	48.39	**63.96**

  - M3E：https://huggingface.co/moka-ai
  - E5（2022，引用190）：https://arxiv.org/pdf/2212.03533
    - https://github.com/microsoft/unilm/tree/master/e5
    - https://huggingface.co/intfloat/e5-large-v2
- 消融实验
  - 训练阶段
    - Table 3: Ablation of the training data, C-MTP, and the training recipe.

model	Dim	Retrieval	STS	Pair CLF	CLF	Re-rank	Cluster	Average
M3E (large)	1024	54.75	50.42	64.30	68.20	59.66	48.88	57.66
OpenAI-Ada-002	1536	52.00	43.35	69.56	64.31	54.28	45.68	53.02
BGE-*pretrain*	1024	63.90	47.71	61.67	**68.59**	60.12	47.73	59.00
BGE w.o. pre-train	1024	62.56	48.06	61.66	67.89	61.25	46.82	58.62
BGE w.o. Instruct	1024	70.55	53.00	76.77	68.58	64.91	**50.01**	63.40
BGE-*finetune*	1024	**71.53**	**54.98**	**78.94**	68.32	**65.11**	48.39	**63.96**

  - batch size
    - **Table 4: Impact of batch size.**

Task \ Batch Size	256	2,048	19,200
Retrieval	57.25	60.96	**63.90**
STS	46.16	46.60	**47.71**
Pair CLF	**62.02**	61.91	61.67
CLF	65.71	67.42	**68.59**
Re-rank	58.59	59.98	**60.12**
Cluster	**49.52**	49.04	47.73
Average	56.43	57.92	**59.00**

4. 启发（可以借鉴的东西）

- 数据清洗可以利用已有的模型来进行蒸馏获取
- 针对脏数据可以先采用无监督方法训练+微调的策略

5. 参考资料

- 论文：http://arxiv.org/pdf/2309.07597
  - code：https://github.com/FlagOpen/FlagEmbedding