Semi-weakly supervised learning for sci-text classification

Zhitong Xu (zx581) zx581@nyu.edu
Computer Science Department
New York University Courant Institute of Mathematical Sciences
New York, NY 10012, USA

Abstract

Given the rapid surge in unlabeled data nowadays, the need for an efficient and accurate labeling method is more important than ever. With more advanced language models(BERT) that published in recent year, labeling text data accurately become plausible. However, a fine-tuned large language model often suffers with high inference time. Our project aims to produce a target model for Sci-text classification with high testing accuracy and low inference time, using limited labeled data and a large scale of unlabeled data.

Keywords: Natural language processing, Text classification, semi supervised learning

1. Introduction

We have already entered big data era, where 2.5 quintillion bytes of data were created everyday and nearly all of them are unlabeled. For most business, they only care about internal meaning of those data. For example, WHO want to keep surveillance on pandemic and they gathered and grouped all meta data for newly founded virus. It would be inefficient and impossible to read and monitor all those meta data for every virus. So label each of those disease with a category (like GHS index) is important.

However labeling them manually is costly and unreliable. Most common approach to discover latent structure is by unsupervised learning. But we all know there is a big gap in accuracy between supervised learning and unsupervised learning. Suppose we want to find the underlying distribution by training a discriminative model, $P(Z|X)$. From Bayes theorem $P(Z|X) = \frac{P(X|Z)\times P(Z)}{P(X)} = \frac{P(X|Z)\times P(Z)}{\sum_{z\in Z} P_Z(z)\times P(X|z)}$. In supervised learning, we already know all possible value that Z can take. Hence we can use a Gaussian to model $P(Z)$, and a DNN for $P(X|Z)$. But in unsupervised learning, Z is not observable, so most common approach is to maximize log likelihood of X by maximizing ELBO, in order to approximate $P_\theta(X|Z)$ and $Q_\phi(Z|X)$.

In this project I primarily focused on text data, since there has been a huge innovation in NLP language model in recent years. With those transformer based model, accuracy has improved significantly if compared with traditional RNN. However, those transformer based model still suffers from $O(T^2d)$ forward passing time. I proposed a semi-weakly supervised learning framework for large scale text categorization that would produce a labeling model with high testing accuracy and low inference time.

2. Prior related work

One semi-weakly supervised learning framework has been proposed by Meta AI in 2019 for large scale image classification(Mahajan). They used weakly supervised learning to pre-train a teacher model, and fine-tune it with limited labeled images. Then distillation among all unlabeled data, then pre-train a student model for production. By this approach, student model shows improvement compared with fully-supervised learning.

Additionally, a pre-trained Bert base has been published in 2019(Beltagy et al., 2019). They trained per-train a Bert base model with large corpus for science document and evaluated its performance on tasks like NER, CLS, and REL. Compare with Bert base, Sci-bert shows better result among tasks for science texts(+3.55 F1 with finetuning and +1.13 F1 without).

While combining those existing work it seems plausible to build a text classification model with high accuracy and low inference time. There still many areas that need to be determine, like what the architecture for production model should be and how to sample from unlabeled data to obtain our manually labeled data set in order to achieve best testing accuracy. The most important question is that, how different choices of teacher model and student model would affect final production model's testing accuracy.

3. Framework & Model

3.1 Framework outline

1. Pre-train BERT on science corpus, treat it as large capacity model.
2. Sample 200 non-cross-listed abstracts from arXiv with its category as labeled data. ($D = \{X1, Y1\}$).
3. Sample 700000 random abstracts as unlabeled data($X2$).
4. Fine-tune large capacity model with labeled data.
5. Make inference on all unlabeled data with fine-tuned large capacity model. $Y2 = P_{Teacher}(Y|X2)$
6. Use top-K sampling to distillate previously inferred data. ($\hat{D} = \{X3, Y3\}|X3 \in X2$, $Y3 \in Y2$)
7. Pre-train target model with previously sampled data. $P_{Student} = P(Y3|X3)$
8. Fine-tune target model with labeled data set.

3.2 Large Capacity model

I used a pre-trained Bert(Sci-Bert) for main block for large capacity model. This served as weakly supervised learning, since Sci-Bert is trained on large science corpus, and our training data is also science text. Hence, by just passing the document into Sci-Bert, we would have a high log likelihood. More formally,

$$Log(P_{Sci-bert}(X)) \approx \arg\max_{\theta} Log(P_{\theta}(X)), X \sim U(arXiv)$$

The input of Bert layer is text of document and we map it to a tensor of integer. The output is representation of CLS token, which is of size 768. Treat it as the encoding for whole document.Then pass it to a two layer MLP in order to set number of output in this model the same as actual number of categories.

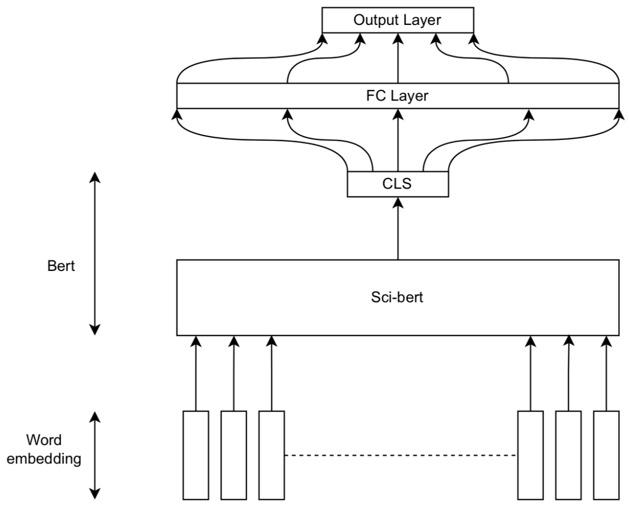

Figure 1: Large capacity model architecture

3.3 Target model

For target model, I used a single layer bidirectional LSTM with attention. Since our task only needs encoder, each state should see information behind it. Hence a bidirectional will be appropriate. To mitigate the effect of vanish gradient problem, I add attention layer after LSTM layer, so all hidden states will contribute to output. Here keys and values are hidden states for each timestamp, and query is the con-cat of end states for both directions. This attention layer act as a weighted sum of all hidden states. Since there is only 1 query, so computation time is $O(Td)$.

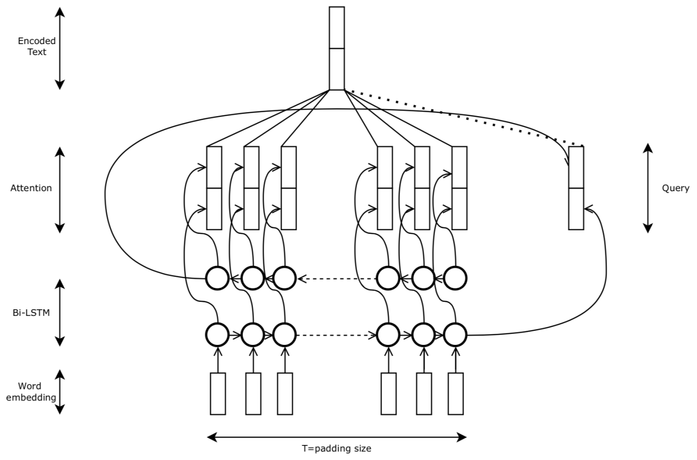

Figure 2: Target model architecture

4. Data sampling process

4.1 Inductive bias

Bert cannot handle large size text due to limit number of position encoding and restrain on computation time, which all input should be less then 512 and LSTM suffers from vanishing gradient problem when corpus are large. The best way is to summarize the whole document into shorter one. Luckily, a document abstract is already a summery for it. Hence, the assumption is that categorize abstract is the same as categorize the full document.

4.2 Labeled data

For labeled data, I sampled 200 non-cross-listed document's abstracts from each category in order to make data set balance. By keeping only non-cross-listed abstracts in our training data set, we can ensure their labels probability are strictly one-hot. Hence we don't need to guess around what is the true probability distribution for each document. The downside of this is that now our sampling process for labeled data and unlabeled data are different, this might affect our testing accuracy.

4.3 Unlabeled data

For unlabeled data, I sampled 70000 random abstract from arXiv.org.

5. Experiment

In Section 3.1, the framework mentioned there used Sci-Bert as large capacity model, and LSTM with attention as target model. I did four experiments with different model selections: 1.Sci-Bert + LSTM, 2.Sci-Bert only, 3.LSTM with only, 4.LSTM + LSTM. (Note: In setup 2 and 3, we treat it as fully supervised learning. In setup 1 and 4, first model is large capacity model, and second model is target model)

5.1 Experiment detail

For fine-tuning of Sci-Bert, I used Adam as optimizer with initial learning rate of 1e-5, and decay of 1e-6. For pre-training of LSTM, I used slightly higher initial learning rate, 1e-3. And for fine-tuning of LSTM, I set learning rate schedule to exponential decay, with initial learning rate as 1e-3, step size equals 1000, and decay rate as 0.96.

5.2 Results

During experiment, I measured target model's top 1 accuracy and inference time on test data. Note: For inference time measurement, I set the largest batch size as my GPU could load.(Batch size=1024)

Model selection	top 1 accuracy	inference time(Sec)
Sci-Bert+LSTM	89.7%	1.36 sec
Sci-Bert	93.4%	23.1 sec
LSTM	71.7%	1.14 sec
LSTM+LSTM	72.9%	1.21 sec

Table 1: Accuracy and inference time for all model selections.

5.3 Encoding visualization

To better understand how well our encoder in target model performs, I took the output from its encoding layer. Here each document has been encoded to a tensor of size 512, and use a auto-encoder and T-sne to reduce to 2D. Note: Color of each data point reflects its true label.

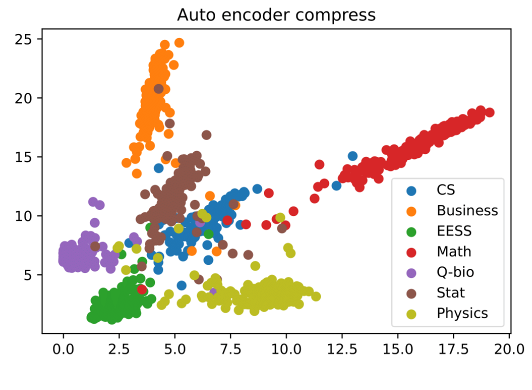

Figure 3: Auto-encoder

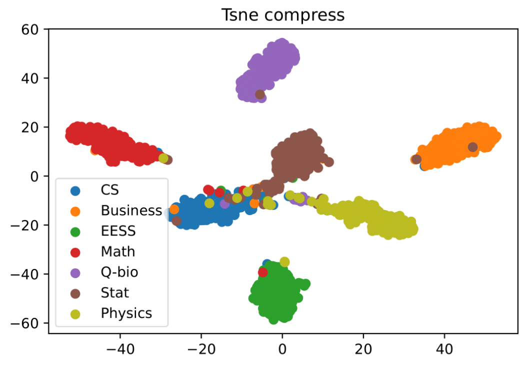

Figure 4: T-sne

6. Analysis

Our result shows that there is a large boost in accuracy for LSTM if we pre-train data inferred from a large capacity model like Sci-Bert. And if the large capacity model has a low accuracy, then accuracy boost will only be slightly. With this training framework, our target model accuracy is coming close to large capacity model, but with significantly reduced inference time. By visualizing our encoder result, we can see a lot of documents wrongly classified as computer science. One plausible explanation would be most other category nowadays uses computer science related approach, especially Statistics.

6.1 Theoretical analysis

Although BERT and other masked language models are not a traditional autoregressive language model like GPT, many recent approach has shown it as a MRF language model. Here I will take the approach shown in Masked Language Model Scoring(Salazar et al., 2019).

Bert used pseudo log likelihood learning, $PLL(\theta; D) = \frac{1}{|D|} \sum_{X \in D} \sum_{t=1}^{|X|} logP(x_t|X_{\setminus t})$, where D is all corpus for training and X is each document in D. $X_{\setminus t}$ is document X, but with [MASK] at position t. Hence, pre-training of Bert is to maximize pseudo log likelihood. However, in traditional auto-regressive models, $logP_{LM}(D) = \sum_{t=1}^{|D|} logP(x_{t+1}|x_{1..t})$. And in [3], they shows empirical result that $PLL(\theta; D) \geq logP_{LM}(D)$. This result demonstrate that by pre-training Bert on D, it will end up with high pseudo log-likelihood for other corpus similar to D.

Now we want to show why fine-tuning a Bert model would yield great encoding for text classification. Suppose after pre-training, we got $P_{Bert}(\bar{x}|\hat{x})$, with maximized PLL. Due to the architecture of transformer, $\bar{x}$ is generated by self-attention, where it get encoded information from all tokens around. Hence, the CLS token will get complete encoding for entire document. Our FC layer then tries to approximate $P_{FC}(Z|CLS)$. Now putting everything together, a fine-tuned model give us:

$$\arg\max_{Z} P_{FC}(Z| \arg\max_{CLS} P_{Bert}(CLS|X - CLS))$$

.

Compare with LSTM, attention based model will have better long-term dependency. Thus Bert will outperforms LSTM. However, due to it's $O(T^2 d)$ computation in self-attention, its inference time will be way slower than LSTM.

7. Conclusion & improvement

This project demonstrated an approach to categorize text with relatively high accuracy and low inference time. The goal is to combine advantages of both models and produce a target model. And in result section, our target model has significantly low inference time with slightly lower accuracy compare with Sci-Bert. There are many other semi supervised approaches like temporal ensembling, mean teacher, and virtual adversarial training. In comparison, this framework has a simpler objective and can combine with model of different architecture.

There are still many improvement that can be made. First, we can set a scheme to generate actual category distribution for each labeled data, instead of a one-hot vector. One naive approach is to assign largest probability to main category and assign the same probability for all other minor categories. Second, for LSTM model, by adding more layers(¡4) would potentially improve its testing accuracy. Lastly, if we ensemble with all RNN like models and use bootstrap aggregating on labeled data, we can create a model with less over-fitting and better generalization. During run-time all model will run parallel, since no all-reduce or communication needed, inference time will not increase compare with non-ensemble model.

References

Iz Beltagy, Arman Cohan, and Kyle Lo. Scibert: Pretrained contextualized embeddings for scientific text. *CoRR*, abs/1903.10676, 2019. URL `http://arxiv.org/abs/1903.10676`.

Dhruv Mahajan. Billion-scale semi-supervised learning for state-of-the-art image and video classification. URL `https://ai.facebook.com/blog/billion-scale-semi-supervised-learning/`.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Pseudolikelihood reranking with masked language models. *CoRR*, abs/1910.14659, 2019. URL `http://arxiv.org/abs/1910.14659`.