CS188 Project Final Report

Christopher Kha, Michelle Zhuang, Miles Kang, Sei Chang

University of California, Los Angeles
{chriskha, michelle.zhuang, milesjk, seichang00}@g.ucla.edu

March 19, 2022

1 Introduction

This project demonstrates an introduction to the fundamental process that goes into modern natural language processing research. We experimented with the training pipeline using two popular common sense reasoning datasets: Sem-Eval-2020 and Com2Sense.

2 Referenced Related Works

The majority of our sources were referred to in the beginning stages of our research, when we wanted to study the fundamentals of transformers and Masked Language Modeling (this was before these concepts were thoroughly taught in lectures). For our Part-of-Speech tagging in the open-ended section, we drew inspiration from prior research on masking policies for parts of speech to mask (Ye et al., 2022) and documentation for the PoS tagging library `nltk` (Bird, Klein and Loper 2022).

3 Main Methods

3.1 Basic Tasks

1. **Data Processing**. We parsed each dataset (`train`, `dev`, `test`) to instantiate objects that would correspond to each row of data. The data processing pipeline was implemented for both *Com2Sense* and *Sem-Eval* datasets.

2. **Model Building**. We selected `bert-base-cased` as the model we use to train for each task. We tuned hyperparameters such as `save_steps` and `logging_steps` to significantly lower the runtime and train more epochs within the limited compute time we had. Each were set to 500.

3. **Model Training and Evaluation**. We completed the implementation for the training loop, which involved computing the loss, performing backward propagation to obtain gradient of the loss, and using an optimizer to update the variables. In the evaluation loop, we used the predicted output to compute performance metrics (accuracy, precision, recall, and f1-score) for each iteration and saved them as checkpoints at regular intervals for reference.

   We also implemented `checkpoint-best` to store the 3 best checkpoints to a file called checkpoint-best based on their accuracy scores. We used the output from `checkpoint-best` to determine which checkpoints to evaluate and submit for our test trials.

4. **Transferring learned knowledge**. We employed a two-stage training tactic by first training our model on the Sem-Eval dataset, then restoring the training pipeline from a checkpoint with the Com2Sense dataset with the hope that the model would learn some "common sense" prior to beginning Com2Sense.

3.2 Open-Ended Task

We aimed to improve our model performance by incorporating Part-of-Speech Tagging to our masking logic. We changed the implementation of our `mask_tokens` function to mask out tokens that correspond to selected parts of speech rather than performing a random sampling of the entire input. By masking more concrete words with PoS such as nouns and verbs, we would allow the transformer model to specifically reconstruct words that are more important to the overall meaning and context of the sequence during training (Bird, Klein and Loper 2022).

3.2.1 Setup

1. **Changes**. For our model pretraining, we modified `mask_tokens` to only random sample from tokens that corresponded to certain parts of speech. The total proportion of word tokens masked would still approximate the original 15 percent of the total.

2. **Technical Implementation**. We incorporated the `nltk` library to convert our list of input tokens into a list of tuples each containing a token and its corresponding PoS tag.

3. **Random Sampling**. We decided to randomly sample words whose tags were prefixed with **NN** (nouns) or **VB** (verbs). We added a new argument `mask_pos` that would accept a regex pattern for determining which tags to sample from.

3.2.2 Integration with NLP Pipeline

1. **Data Processing**. The data processing pipeline implemented for both Com2Sense and Sem-Eval datasets remained the same. The key difference was (1) accepting a new user optional parameter that would determine the PoS sampled for masking and (2) changing the `mask_tokens` to create a new tensor whose elements would correspond each token to the PoS tag.

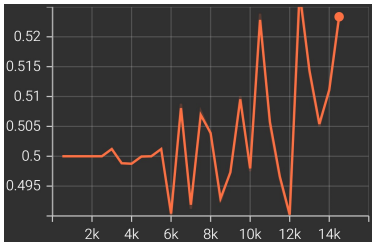
2. **Model Building**. We would first pretrain with MLM on the semeval dataset. Then we would finetune semeval using the MLM pretrained model as the intermediary stage to our com2sense training. We also used the pretrained model `bert-large-cased-whole-word-masking`, which is based on the original `bert-large-cased` but pretrained with the Whole Word Masking strategy. To perform transfer learning from Sem-Eval-2020, we selected the model that performed the best on the `dev.csv` set.

3. **Model Training and Evaluation**. Our training and evaluation methods remained the same. We set `save_steps` and `logging_steps` to 500 for consistency between training runs.

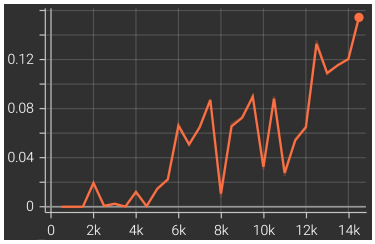
4 Main Results

Checkpoint Link: https://tinyurl.com/6j6ffjmv

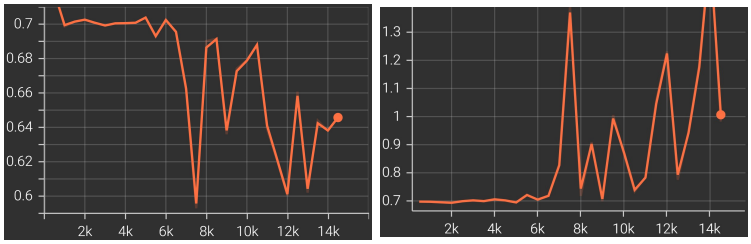
4.1 Supervised Learning



Supervised Accuracy



Supervised Pairwise Accuracy

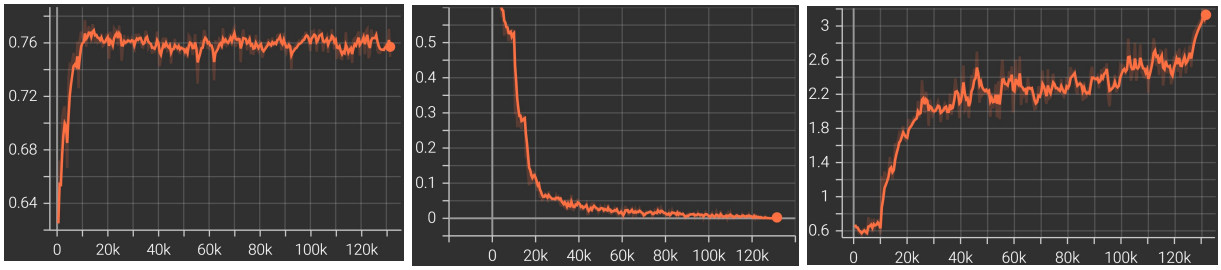


Supervised Loss Supervised Eval Loss

Figure 1. Accuracy, pairwise accuracy, training loss, and evaluation loss on `dev.json` dataset for Com2Sense Supervised Learning across 14500 learning iterations.

The results for supervised learning in all metrics oscillate a sizeable amount. The oscillations could results from a high learning rate which made the model overcompensate for the errors in the previous step, which is shown in the training loss curve. Nevertheless, we can observe that the loss curve still decreases continuously which indicates that the training was still successful.

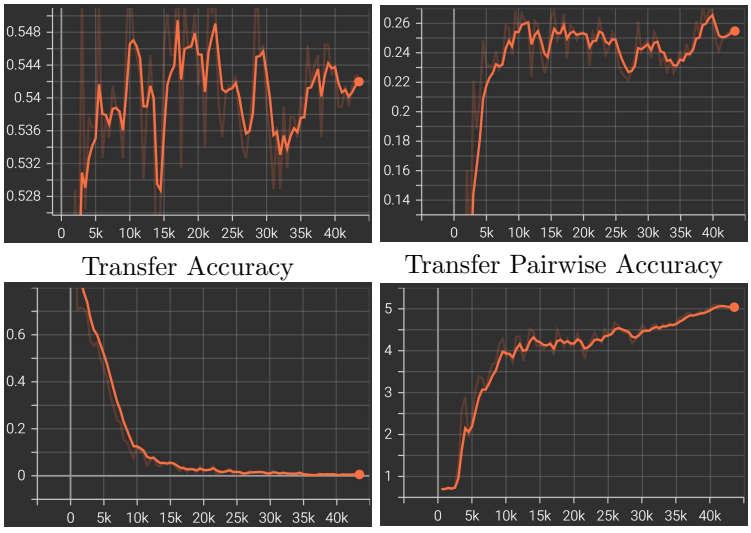
4.2 Knowledge Transfer



SemEval Accuracy SemEval Loss SemEval Eval Loss

Figure 2. Accuracy, training loss, and evaluation loss on `dev.csv` dataset for SemEval 2020 Task 4 across 130000 learning iterations.

The results above demonstrate that *Sem-Eval-2020 Task4* proves to be the easier task to finetune directly on. The model is able to achieve a much higher accuracy and demonstrate better learning curves than on the *Com2Sense* task. We are able to reach a peak accuracy of approximately 0.78, which is far beyond the 0.523 peak accuracy reached for the regular supervised task.



Transfer Accuracy Transfer Pairwise Accuracy

Transfer Loss Transfer Eval Loss

Figure 3. Accuracy, pairwise accuracy, training loss, and evaluation loss on `dev.json` dataset for knowledge-transferred Com2Sense Supervised Learning across 43500 learning iterations.

Finetuning on our most successful model directly trained on the *Sem-Eval-2020 Task4* dataset (iteration 122500), we were able to achieve much improved results compared to directly finetuning on *Com2Sense*. We average at around 0.54 overall accuracy and peak at 0.555. The improvements are even more drastic for pairwise, which now averages at around 0.26 accuracy.

4.3 Other Analysis

domains	one-staged	two-staged	open-ended
time	0.523	0.567	0.588
physical	0.530	0.590	0.552
social	0.534	0.507	0.545

Table 1. Accuracy on on *Com2Sense* `dev.json` for one-staged, two-staged, and open-ended models across time, physical, and social domains.

scenarios	one-staged	two-staged	open-ended
comparison	0.541	0.579	0.596
causal	0.517	0.532	0.527

Table 2. Accuracy on *Com2Sense* `dev.json` for one-staged, two-staged, and open-ended supervised learning across comparison and causual scenarios.

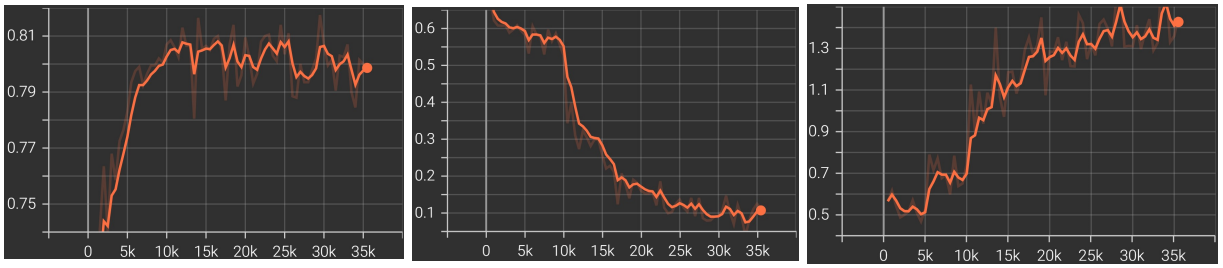
numeracy	one-staged	two-staged	open-ended
True	0.510	0.541	0.551
False	0.540	0.564	0.568

Table 3. Accuracy on *Com2Sense* `dev.json` for one-staged, two-staged, and open-ended supervised learning across sequences that did or did not require numeracy.

We observe that the two-staged model performs much better than the one-staged model, and our model with PoS tagging (open-ended model)

4.4 Open-Ended Task

4.5 Knowledge Transfer



SemEval Accuracy SemEval Loss SemEval Eval Loss

Figure 4. Improved accuracy, training loss, and evaluation loss on `dev.csv` dataset for SemEval-2020 355000 learning iterations using pretrained model.

We can observe that the results from *SemEval-2020* are improved over the previous iteration. The accuracy levels have increased to a peak of 0.817 while averaging at around 0.8 in the later iterations. The evaluation loss is also more than halved compared to that of the regular transfer learning processing. The best results came from iteration 29500.

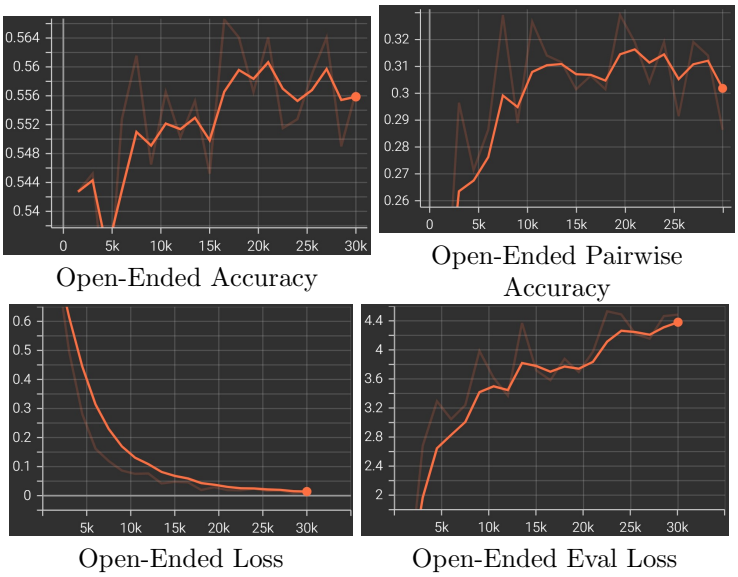


Figure 5. Accuracy, pairwise accuracy, training loss, and evaluation loss on `dev.json` dataset for knowledge-transferred Com2Sense Supervised Learning across 30000 learning iterations.

Finetuning on our most successful results on the Sem-Eval (iteration 29500), we were able to achieve improved results compared to those produced through regular transfer learning on *Com2Sense*.

5 Conclusion and/or Discussions

Performance comparisons between runs: Unsurprisingly, performance correlated generally well with the complexity of the model; our regular supervised training yielded the lowest accuracy, our transfer learned training yielded better accuracy, and our transfer learned training with PoS tagging had the best accuracy of about 0.817 on Sem-Eval and 0.56 on Com2Sense.
Overfitting?: The increasing eval loss on the data, despite training loss continuously decreasing, indicates that our model begins to overfit, meaning that as it starts to get better at predictions for the data it can see, it gets worse and worse at predictions for data it cannot see. We also notice that on all our trials, the upward trend in accuracy eventually stagnates at a similar point to when the eval loss starts to skyrocket, so we can infer that this point is when the overfitting begins to happen.
Takeaways: The Masked Language Model is an effective language task that can predict masked tokens correctly the majority of the time given sufficient training. It does start to overfit and lose effectivity over time. Com2Sense yields higher accuracy than Sem-Eval at the cost of additional computation for complementary statements and domain and scenario categorization. Our results suggested that predictions in the temporal domain were easiest to train. Finally, our masked language model can be improved with Part-of-Speech tagging, which gives the model a more targeted and content-focused approach in its training.

References

[1] Steven Bird, Ewan Klein, and Edward Loper. Categorizing and tagging words, 2019. https://www.nltk.org/book/ch05.html, Last seen 2022.

[2] James Briggs. Training bert #1 - masked-language modeling (mlm), 2021. https://www.youtube.com/watch?v=q9NS5WpfkrU, Last seen 2022.

[3] James Briggs. Masked-language modeling with bert, 2021. https://towardsdatascience.com/masked-language-modelling-with-bert-7d49793e5d2c, Last seen 2022.

[4] Rani Horev. Bert explained: State of the art language model for nlp, 2018. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270, Last seen 2022.

[5] Karthikeyan K. An out-and-out view of transformer architecture, 2021. https://medium.com/mlearning-ai/an-out-and-out-view-of-transformer-architecture-6926da4c8080, Last seen 2022.

[6] Cathal Horan. Unmasking bert: The key to transformer model performance, 2021. https://neptune.ai/blog/unmasking-bert-transformer-model-performance, Last seen 2022.

[7] Qinyuan Ye, Belinda Li, Sinong Wang, and Hao Ma Wen-tau Yih Xiang Ren Madian Khabsa Bolte, Benjamin. On the influence of masking policies in intermediate pre-training, 2019. https://aclanthology.org/2021.emnlp-main.573.pdf, Last seen 2022.