Natural Language Processing Project

Michele Sanna

This project is divided in two parts. The first part involves the fine tuning of two language models (BERT and GPT-2) for a sentiment analysis task. The second part consists in an examination of the hidden layers of the base models (intended as without any fine tuning) to see if the sentiment of a sentence is in some way encoded in the hidden states of a language model even without any fine tuning

1 Fine tuning

This section will discuss the methodology of the fine tuning on a sentiment analysis task of the two models used in this project: Google's BERT (Devlin et al., 2018) and OpenAI's GPT-2 (Radford et al., 2019)

1.1 Models

For both GPT-2 and BERT has been used the smallest model's version available. The number of parameters of these two versions are comparable (around 110 million for BERT and around 117 million for GPT-2). To make the models more comparable, for BERT has been used the cased version, because GPT-2 comes only with cased versions

1.2 Dataset

The dataset used is a collection of Amazon's products reviews (Ni et al., 2019) in which all users and items have at least 5 reviews. The dataset contains a lot of information for every review, but for the aim of this project only the review's text and the overall vote have been used. Moreover, because the dataset is very large, only a subset of reviews has been selected, in such a way that every overall vote value is equally represented (precisely one million reviews for every overall vote value) and the length of each sentence is between 200 and 510 tokens.

1.3 Training methodology

The training is structured as a classification task, where every review's overall vote (an integer ranging from 1 to 5) is interpreted as a class to be predicted using the tokenized review as the input for the model. According to that, the loss function used is the cross entropy loss.

The optimizer used is the AdamW optimizer, with an initial learning rate of 5e-5 as recommended both in BERT (Devlin et al., 2018) and GPT-2 (Radford et al., 2019) papers. Even if AdamW is an adaptive optimizer, it's also been used a learning rate scheduler, precisely a cosine annealing scheduler with warm restarts, because the AdamW paper (Loshchilov and Hutter, 2017) itself shows that it's the optimal choice.

In order to train the models with the whole dataset in a reasonable time the training was distributed across two nodes, each with two Nvidia V100 GPU's. For the same reason the first half of the model's weights was frozen both for BERT an GPT-2, this paper (Lee et al., 2019) shows that this should not affect the quality of the results in a relevant way.

For each model were done two trainings, one of 5 epochs to obtain the best result, and one of only one epoch with 6 warm restarts to observe the convergence speed during the training.

1.4 Results and discussion

5 epochs training

	Test Loss	Test Accuracy
BERT	0.9349	0.6046
GPT-2	0.9250	0.6036

Table 1: Summary of the 5 epochs training

After the long training both models achieved an accuracy around 0.6. Probably it wasn't possible to obtain a significantly higher accuracy because of the structure of the dataset, indeed it's often difficult even for a human to tell the number of stars of a review from its text (from example it's hard to tell apart a review of 3 stars from a review of 4)

Single epoch training

	Test Loss	Test Accuracy
BERT	0.9655	0.5849
GPT-2	0.9432	0.5935

Table 2: Summary of the single epoch training

In the single epoch training GPT-2 converged faster, even if during the training BERT had a smaller average loss.

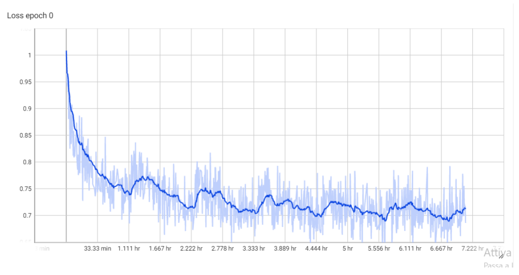

Figure 1: BERT loss during training (smoothed)

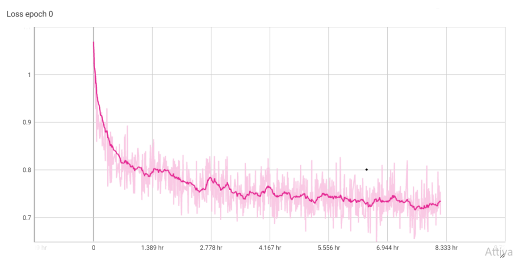

Figure 2: GPT-2 loss during training (smoothed)

2 Linear probes

The aim of this section is to check if, even the untrained model (intended as without any fine tuning, but with the maskedLM or causalLM weights), produces internal representations that encode the sentiment of a sentence, and secondly if the quality of this encoding for a sentence is affected by the perplexity of the model.

2.1 Methodology

To assess if an hidden state encodes the sentiment of a sentence, we train a single linear classifier in order to predict the sentiment of the sentence using as input only the representations of that hidden state; exploiting the idea of "linear probe" (Alain and Bengio, 2016). In order to do that the problem used for the fine tuning task has been simplified, reducing the class number from 5 to 2, using only reviews that have either 5 stars or 1 star. For each hidden state of each model two linear probes have been trained: one for low perplexity sentences and one for high perplexity sentences.

2.2 Perplexity

To establish how much the model "understands" a sentence (intended as how well it predicts the words of that sentence) in the case of GPT-2 we use the perplexity metric, defined as:

$$Perplexity(X) = exp\{-\frac{1}{L}\sum_{L}^{i} \log p_\theta(x_i|x_{<i})\}$$

Where $X = (x_0, x_1, ..., x_L)$ is the tokenized sentence.

Because BERT is bidirectional the perplexity isn't the best suited metric for it, so in the case of BERT we use a pseudoperplexity metric, previously used in (Lin et al., 2023), defined as:

$$PseudoPerplexity(X) = exp\{-\frac{1}{L}\sum_{L}^{i} \log p_\theta(x_i|x_{j\neq i})\}$$

For our experiment the low perplexity sentences are the ones which have a perplexity (or pseudoperplexity in the case of bert) under the 12.5° percentile, the high perplexity sentences instead are those with a perplexity over the 87.5° percentile.

2.3 Results and discussion

2.3.1 GPT-2

	Low Perplexity		High Perplexity	
	Loss	Accuracy	Loss	Accuracy
Layer 0	0.6712	0.572	0.6861	0.5195
Layer 1	0.5080	0.765	0.5708	0.71975
Layer 2	0.4777	0.79	0.5498	0.73125
Layer 3	0.4749	0.7875	0.5559	0.72675
Layer 4	0.4663	0.7907	0.5368	0.744
Layer 5	0.4438	0.7982	0.5082	0.761
Layer 6	0.4246	0.8135	0.4866	0.7745
Layer 7	0.4253	0.813	0.4756	0.7777
Layer 8	0.4275	0.8207	0.4719	0.7785
Layer 9	0.4405	0.8102	0.4808	0.7755
Layer 10	0.4913	0.788	0.4844	0.7707
Layer 11	0.5039	0.7852	0.4815	0.7685
Layer 12	0.4653	0.7837	0.4879	0.7587

Table 3: Performance of each layer's linear probe. Layer 0 is intended as the embedding layer

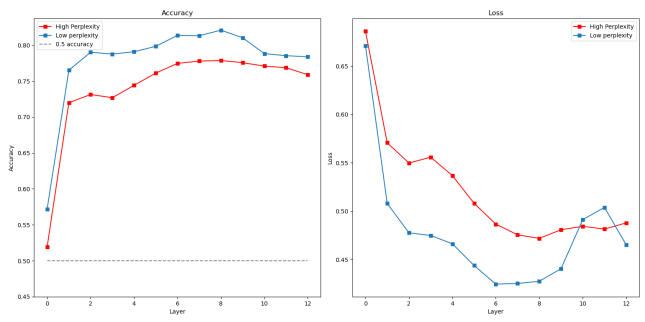

Figure 3: Table 3 as plots

We can observe that every hidden state encodes in some way the sentiment of a sentence, since the accuracy of every linear probe is over 0.5. The quality of this representation tends to peak around layer 7-8, and the linear probes of the successive layers are less accurate. Probably this is due to the higher specialization of the last layers for the task of next word prediction, indeed the last layer in particular is encoding a representation that has to be translated into a distribution of probability over words. We also can observe that the perplexity of the sentence influences the accuracy of the prediction, and so the quality of the internal representations.

2.3.2 BERT

	Low Perplexity		High Perplexity	
	Loss	Accuracy	Loss	Accuracy
Layer 0	0.6931	0.5	0.6933	0.5
Layer 1	0.6804	0.5785	0.6671	0.611
Layer 2	0.6715	0.5825	0.6273	0.663
Layer 3	0.6683	0.61225	0.5809	0.6902
Layer 4	0.7814	0.583	0.5922	0.6787
Layer 5	0.8020	0.583	0.5885	0.6847
Layer 6	0.7551	0.5945	0.5824	0.6925
Layer 7	0.7576	0.59325	0.5595	0.704
Layer 8	0.7265	0.6085	0.5614	0.709
Layer 9	0.6612	0.6492	0.5127	0.7447
Layer 10	0.5365	0.7387	0.4313	0.8065
Layer 11	0.5443	0.7387	0.4192	0.8107
Layer 12	0.5435	0.7375	0.4368	0.798

Table 4: Performance of each layer's linear probe. Layer 0 is intended as the embedding layer

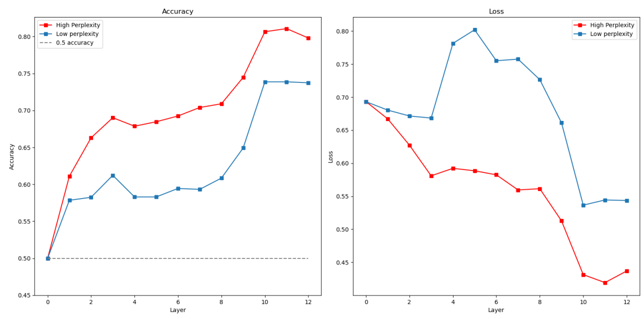

Figure 4: Table 4 as plots

The results of BERT are pretty different from the results of GPT-2. First of all, the embedding layer can't encode the sentiment of a sentence (its accuracy indeed is 0.5). That's due to the fact that for GPT-2 the token that we're using for prediction is the last token, that represent an actual word of the sentence, while for BERT we're using the first token, that is a special token (CLS token) used for classification tasks.

Secondly, the peak accuracy is around layer 10-11, much more further on the network with respect to GPT-2. Probably that's because in the pre-training of BERT there was an actual classification task (the next sentence recognition task), so the last layers are less specialized in predicting a word with respect to the last layers of GPT-2.

Lastly, for BERT the intuition of the inverse correlation between perplexity and accuracy doesn't seem to hold.

References

Alain, Guillaume and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Lee, Jaejun, Raphael Tang, and Jimmy Lin. 2019. What would elsa do? freezing layers during transformer fine-tuning.

Lin, Zeming, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. 2023. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130.

Loshchilov, Ilya and Frank Hutter. 2017. Decoupled weight decay regularization.

Ni, Jianmo, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. pages 188–197.

Radford, Alec, Jeff Wu, Rewon Child, D. Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.