SHAZAM: The Effects of Pre-Training and Fine-Tuning on Cross-Domain Sentiment Analysis With ELECTRA

Silvia Gong
Stanford University
silvgong@stanford.edu

Anfal Siddiqui
Stanford University
anfal@stanford.edu

Meredith Xu
Stanford University
mereyxu@stanford.edu

Abstract

Sentiment Classification is an active area of research within the Natural Language Processing community, particularly the more difficult ternary formulation that attempts to classify text as positive, negative, or neutral. Recent works have shown that the BERT model benefits from further pre-training as well as fine-tuning on cross-domain sentiment classification (Rietzler et al., 2020) (Gururangan et al., 2020) (Sun et al., 2019). We did extensive experiments on in-domain and cross-domain fine-tuning on SST, Yelp, and Amazon dataset. We discovered that even a little bit of in-domain data during fine-tuning can boost the model's accuracy to near exclusive in-domain fine-tuning. We observed that both in-domain and cross-domain further pre-training hurt ELECTRA's performance while boosting BERT's performance, and hypothesized the root causes of ELECTRA's unsuccessful further pre-training.

1 Introduction

Sentiments are heavily embedded in every English sentence. They indicate the attitudes, feelings, and emotions of the speaker. Supervised Sentiment Classification refers to the task of detecting sentiment polarity in sentences. It is an active area of research within the Natural Language Processing community, particularly the more difficult ternary formulation that attempts to classify text as positive, negative, or neutral. Recent work has shown that sentiment classifiers that are fine-tuned from BERT models can achieve state-of-the-art results on various datasets (Rietzler et al., 2020) (Sun et al., 2019) (Du et al., 2020a) (Du et al., 2020b). These works have also provided guidance on how to best fine-tune BERT models for sentiment analysis across different domains, focusing on the benefits of further pre-training the models using domain-specific text and fine-tuning using joint-domain training (datasets from multiple domains, instead of just one) to achieve their results. However, these prior works used the original BERT models (Devlin et al., 2019) and focused on only two domains, leaving an open question as to whether these techniques generalize to newer BERT-like models (with different pre-training objectives), across a larger number of domains (where there may be semantic clashes among the domains), and when the computational budget is much smaller (fine-tuning on the order of hours, as opposed to days). Our goal is to answer these questions using an ELECTRA model (Clark et al., 2020) and using datasets from the movie, Yelp, and Amazon product review domains. Our approach is to 1). Take the pre-trained ELECTRA small model and perform further pre-training for a small number of steps (i.e. $\leq$ 20,000) using sentiment-laden text. 2). Perform fine-tuning on both out-of-the-box ELECTRA and further pre-trained ELECTRA with data from multiple domains. We hypothesize that further pre-training and joint fine-tuning using all three datasets would offer a greater performance boost over out-of-the-box ELECTRA across all of our evaluation sets than no further pre-training or fine-tuning using only one or two of the datasets. By comparing the results from our further pre-training and fine-tuning experiments using different domain combinations of training data, we found that further pre-training for a small number of steps worsens ELECTRA's performance and having data from all the domains during fune-tuning, even with small amounts, can result in a boost in overall performance across all datasets.

2 Related Work

2.1 Further Pre-Training

Further pre-training has been studied extensively using several variants of BERT. Gururangan et

al. examined the performance gains of domain-specific pre-training with RoBERTa. They focused on two forms of the additional pre-training: domain-adaptive pre-training (DAPT), which uses text that comes from the same general domain as the target task, and task-adaptive pre-training (TAPT), which uses the task's unlabeled data (and therefore comprises a much narrower scope than that of the entire domain and is much less resource intensive) (Gururangan et al., 2020). The authors found both forms of further pre-training to be varying degrees of beneficial across all domains they tested, with the more efficient but targeted TAPT even matching the performance of the broader DAPT for some domains.

As part of their exploration of how to best fine-tune models for text classification, Sun et al. explored the benefits of further pre-training on sentiment analysis (Sun et al., 2019) using a vanilla BERT model. They also found that the further pre-training using specific in-domain data was helpful for the sentiment analysis task, but that pre-training using data from across domains, even if both were sentiment-laden text as found in reviews, was not as helpful as using the domains alone. They particularly focused on cross-domain pre-training using movie and restaurant reviews and suspected the data distributions were too different.

A shortcoming of these prior works on further pre-training is that they focus exclusively on BERT models that use the traditional masked language modeling objective, leaving an open question as to whether models such as ELECTRA and XLNet which use different objectives would also derive benefits from it.

2.2 Cross-Domain Sentiment Analysis

Rietzler et al. focus on improving aspect-target sentiment classification using numerous strategies, including cross-domain further pre-training and fine-tuning, working with laptop and restaurant reviews and off BERT-base. They found that further pre-training using sentiment-laden text, even if it does not match the exact domain as the test set, was still beneficial. They theorize that because the BERT-base model was pre-trained primarily on fact-based text akin to Wikipedia, further pre-training it with any opinion-based sentiment text helps performance on ATSC, irrespective of whether the domains align (Rietzler et al., 2020). They yielded their best performance on both domain test sets by combining their training datasets in a joint fine-tuning approach. They speculated that because the two domains were not in conflict with each other, simply adding them together just provided more training data and led to a standard boost in performance. This left an open-question as to whether the joint-training benefit they observed would have remained had they kept the size of the training set fixed and sampled from both domains, rather than just concatenating them together.

Du et al. tackled cross-domain analysis by introducing novel pre- and post- training procedures to BERT, with the goal of being able to only train using labeled sentiment data in a source domain but have robust performance in the target domain (Du et al., 2020a). First, they replaced the next sentence prediction task with a domain distinguish task, where the model must learn to determine if two input sentences are drawn from a target domain or mixed domains. Next, the MLM objective is run using unlabeled text from the target domain in order to encourages BERT to learn representations for fine-grained opinion words in the target-domain. Finally, the resultant model is put through adversarial training for an aggressive fine-tuning. The authors designed a sentiment classifier and a domain discriminator that operate on the CLS token's hidden state. The sentiment classifier layer is trained using labeled data from the source domain, while the domain discriminator is jointly trained to determine which domain sample text comes from. The BERT model's parameters are optimized to increase the discriminator loss. Du et al. achieved success with this multi-step approach, achieving state-of-the-art results on all Amazon review benchmarks. However, adversarial training on text is known to be notoriously difficult (Clark et al., 2020), so the accessibility of using these methods in a time-efficient manner is unclear.

3 Data

```
Sample review from SST-Tree:
[positive], elaborate continuation

Sample review from Yelp:
[neutral], Its an OK.... Joint.
Good service. The bartender is
really nice and fast. And the
menu is good for quick late night
apps. The karaoke rooms need to
be all speuced up. But would come
```

```
back again!

Sample review from Amazon:
[negative], Would not talk to my
computer nor my smart phone;
sorry!
```

We use the latest version of the Yelp Academic Dataset[1] for further pre-training of ELECTRA Small. It consists of 8,635,403 business reviews, of which we have randomly sampled 100,000 for use in our constrained pre-training setup.

The first dataset we use for fine-tuning and evaluation is the Stanford Sentiment Treebank dataset (SST) (Socher et al., 2013), particularly the SST-3 formulation that uses positive, negative, and neutral labels. It consists of 8,544 movie reviews, which are split up into 159,274 labeled phrases.

The Yelp dataset we use for fine-tuning and evaluation is the same as the one used in (Potts et al., 2020), which is cited from (Zhang et al., 2015). It is derived from an earlier version of the Yelp Academic Dataset. The training file contains 650,000 reviews and their ratings. The test file contains 50,000 reviews and we split it in half by line numbers to create dev and test sets. We label the reviews by putting ratings less than 3 as negative, those equal to 3 as neutral, and those greater than 3 as positive. This formulation follows the one described in (Potts et al., 2020). After processing, there are 260,000 positive examples, 260,000 negative examples, and 130,000 neutral ones in the training set and we took only the fist 100,000 examples for fine-tuning on the Yelp dataset (more details can be found in the Experiments section). There are 9,577 positive examples, 10,222 negative examples, and 5,201 neutral ones in the dev set.

We also use the Amazon review dataset (Ni et al., 2019) because it is a widely-adopted benchmark for sentiment analysis, and it is used in fine-tuning and evaluating models such as BERT-DAAT(Du et al., 2020a), WTN (Du et al., 2020b), and Sentix (Zhou et al., 2020). This dataset contains 233.1 million Amazon product reviews across 29 different categories from May 1996 to Oct 2018. We sample 25,000 recent reviews from each of the four most popular categories: Books, Movies and TV, Electronics, and Home and Kitchen. Specifically, we sampled 8,333 most recent 4/5-star reviews as positive examples, 8,333 most recent 3-star reviews as neutral examples, and 8,334 most recent 1/2-star reviews as negative examples. In the end, our training set consists of 100,000 reviews evenly split between positive, neutral, and negative. We used the same processing technique to obtain our 25,000 dev and test sets.

[1] https://www.yelp.com/dataset

4 Models

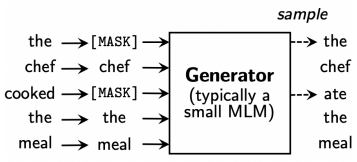

Figure 1: Generator takes in the masked input and replaces the *[MASK]* tokens with learned tokens that resemble original identities.

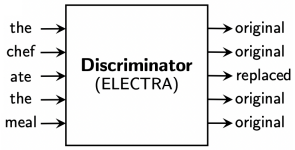

Figure 2: Discriminator takes in the output token sequence of the Generator and predicts whether each token is original or replaced.

ELECTRA ELECTRA (Clark et al., 2020) is a new pre-training approach done to the underlying BERT model that trains the generator and the discriminator transformer models. After randomly replacing tokens in the input by the *[MASK]* token using BERT's Masked language modeling (MLM), the generator replaces the *[MASK]* tokens with alternative tokens sampled from a small generator network in order to recreate the original tokens. Specifically, it is a small MLM that is trained jointly with the discriminator using the maximum likelihood objective. If the generator happens to predict the same token as the masked original, the predicted token is treated as 'original' instead of 'replaced'. Let $x = [x_1, ..., x_n]$ be the sequence of input tokens. The generator's encoder maps x into contextualized vector representations $h(x) = [h_1, ..., h_n]$. Let t be the position where $x_t = [MASK]$, and let e be token embeddings. The generator outputs an output distribution over token x_t with a softmax layer:

$$p_G(x_t|x) = \frac{exp(e(x_t)^T h_g(x)_t)}{\sum_{x'} exp(e(x')^T h_g(x)_t)}$$

The other part of the model is the discriminator whose goal is to predict whether a token in its input

sequence is sampled from the generator network or part of the original input using a sigmoid output layer:

$$D(x,t) = sigmoid(w^T h_D(x)_t)$$

The loss equations are as follows:

$$\begin{aligned} L_{MLM}(x,\theta_G) =& \mathbb{E}(\sum_{i\in m} -logp_G(x_i|x^{masked})) \\ L_{Disc}(x,\theta_D) =& \mathbb{E}(\sum_{t=1}^{n} -\mathbb{I}(x_t^{corrupt} = x_t) \\ & \cdot logD(x^{corrupt},t) \\ & -\mathbb{I}(x_t^{corrupt} \neq x_t) \\ & \cdot log(1-D(x^{corrupt},t))) \end{aligned}$$

The combined loss is minimized over a large corpus X of raw text by:

$$L = min_{\theta_G,\theta_D} \sum_{x\in X} L_{MLM}(x,\theta_G) + \lambda L_{Disc}(x,\theta_D)$$

During pre-training, the generator is trained jointly with the discriminator, but only the discriminator is used for fine-tuning for the downstream tasks. Although ELECTRA has a structure similar to GAN, the generator is trained cooperatively instead of adversarially because it is impossible to backpropagate through sampling from the generator. ELECTRA also separates the embedding size from the hidden size, and adds an additional linear layer to project the embeddings from their embedding size to the hidden size if the embedding size is smaller.

Baseline Our baseline is a pre-trained ELECTRA small with a linear classifier head attached on top of the [CLS] token, without additional pre-training or fine-tuning.

Proposed model A recurring theme throughout the papers (Sun et al., 2019) (Rietzler et al., 2020) is that additional domain-specific pre-training of BERT models provides a significant boost in performance on the downstream task. BERT models were pre-trained using BooksCorpus and English Wikipedia datasets that are largely factual and informative. The masked word prediction is therefore geared more towards predicting words without emotion or opinion, which influences how it performs as a language model. On the other hand, the sentiment analysis task involves informal and subjective reviews written by the general population. It would not be surprising that further pre-training with sentiment texts would boost the model's performance on downstream sentiment classification tasks.

Since ELECTRA has a nearly identical underlying model structure as BERT, was trained with the same data, and only differs in its pre-training approach, we theorize that the same pre-training rules outlined above should boost ELECTRA's performance as well.

Base on the superior performance of ELECTRA on several benchmarks with smaller model sizes and less training resources, we propose to further pre-train the ELECTRA small with opinion-based sentiment text dataset from Yelp to nudge it towards making more sentiment-esque predictions. We also fine-tuned the same model with different datasets to see how additional out-of-domain fine-tuning could influence its performance. We explored the performance change with only fine-tuning on ELECTRA small, and the combination of both additional pre-training and fine-tuning on both BERT based uncased and ELECTRA small.

5 Experiments

ELECTRA Pre-training We further pre-trained ELECTRA from the released ELECTRA small checkpoint using a random sample of 100,000 reviews from the Yelp Academic Dataset. We used a batch size of 64 and a learning rate of $5e^{-5}$ and ran the pre-training up to 17,000 steps/iterations, saving checkpoints every 1,000 steps.

BERT Pre-training Similarly, we performed additional pre-training on BERT based uncased model with the same 100,000 data points from the Yelp Academic Dataset to compare against the results produced by ELECTRA for our setup.

ELECTRA Fine-tuning We performed fine-tuning on both out-of-the-box ELECTRA and further pre-trained ELECTRA. The further pre-trained ELECTRA is only fine-tuned on the Yelp dataset and we had multiple experiments for fine-tuning out-of-the-box ELECTRA using different combinations of training data. For every fine-tuning experiment we did, we kept the total number of training examples as 100,000. For fine-tuning only on the Yelp dataset, we took the first 100,000 examples in the training set. The Amazon dataset already

has 100,000 training samples so we just shuffled the dataset before passing it into the model. For fine-tuning on multiple datasets combined, we randomly sampled examples from each dataset evenly so the total number of examples is still 100,000. We fixed the random state every time we sampled for reproducibility purposes. In particular, during training, we carved out 10% of the training data for early stopping checks. If the training error doesn't improve for 2 iterations, then we stop the training. After a hyperparameter search, we found a batch size of 16, gradient accumulation steps of 128, and learning rate of 0.0001 to offer the best results. The number of epochs for each experiment is shown in table 1. After each fine-tuning, we evaluated the model on all three dev sets.

5.1 Metrics

Our main metric is the Macro F1 Score (averaged across the datasets) and we use it to compare the performance of different models. We chose Macro F1 because it puts equal weights on different classes. In our datasets, the number of examples from different classes varies and we did not want that variance to affect the overall F1 score across the whole dataset. Furthermore, F1 score takes into account both precision and recall and is widely used in NLP. We also compute precision and recall when evaluating the models so we can get a more nuanced view of the types of errors our model is making than we would using simple accuracy. We calculate F1, precision, and recall for individual classes as well, which allows us to see if the models performs particularly well or poorly at predicting certain labels.

6 Results and Analysis

6.1 Results

See Table 1, 2 for model details. See Table 3 for results on SST, Yelp and Amazon datasets. The values in the tables are scaled by 100.

6.2 The Effects of Further Pre-training

As shown in table 2, the Macro-F1 score after fine-tuning on Yelp dropped from 68.1 for the vanilla ELECTRA small to 59.5 when we performed further pre-training for 10,000 steps. In particular, the score got lower on every dev set of the three datasets, with the most significant decrease on the Amazon dataset. We also noticed similarly poor results for other numbers of pre-training steps (3,000 steps to 17,000 steps).

#	Fine-tuned Dataset(s)	# Epoch	Macro F1
0	/	/	20.2
1	SST	9	57.9
2	Yelp	12	68.1
3	Amzn	14	67.0
4	SST + Yelp	8	68.6
5	SST + Amzn	12	71.1
6	Yelp + Amzn	13	69.8
7	**SST + Yelp + Amzn**	**10**	**73.6**

Table 1: The same vanilla ELECTRA-small model fine-tuned on 100,000 samples randomly drawn from one or more datasets, the number of Epochs it takes to converge, and the Mean Macro F1 score (%) obtained from validating on SST, Yelp and Amazon's dev set. Note that model 0 is the baseline with no pre-train or fine-tune.

#	Model	Pre-train	# steps	Macro F1
8	BERT			55.1
9	BERT	Yelp	10k	55.5
10	BERT	Yelp	17k	55.6
2	**ELECTRA**			**68.1**
11	ELECTRA	Yelp	10k	59.5

Table 2: The Mean Macro F1 score (%) for the BERT-based uncased and the ELECTRA-small models with or without pre-training on Yelp with different number of steps and fine-tuned on Yelp. Note that model # 2 already exists in table 1.

To investigate the reason behind ELECTRA's poor performance with further pre-training, we repeated our pre-training experiment on BERT-base to determine if the issue was with our pre-training setup or something inherent with ELECTRA. BERT has been widely studied for various NLP tasks, has been successfully further pre-trained in prior work, and has a more stable and well documented codebase. We used the exact same pre-training data we used for ELECTRA pre-training on BERT-base and also fine-tuned the further pre-trained BERT model on Yelp. As shown in 2, further pre-training BERT, even for a limited number of steps, did boost the model's performance, albeit modestly. The macro-F1 score of 55.6 after 17,000 steps of pre-training is higher than the 55.1 gained by only fine-tuning the model. The performance gain is most prominent on the Yelp and Amazon datasets, but somewhat less stable for SST. Still, they demonstrate the correctness of our pre-training setup.

Dataset		SST DEV												TEST
Model		0	1	2	3	4	5	6	7	8	9	10	11	7
	R	02.3	86.9	66.4	60.1	87.8	82.4	58.8	83.3	52.1	54.0	37.5	47.7	85.7
+	P	35.7	78.3	73.2	74.6	75.9	75.3	77.2	76.4	59.8	59.9	64.3	59.1	81.1
	F1	04.2	82.4	69.7	66.6	81.4	78.7	66.8	79.7	55.7	56.8	47.4	52.8	83.3
	R	00.0	22.3	27.5	50.7	22.3	20.1	51.1	48.5	13.9	13.1	16.5	29.3	44.2
o	P	00.0	54.3	26.6	25.2	49.0	35.9	24.0	38.5	22.6	22.9	22.9	23.8	31.4
	F1	00.0	31.6	27.0	33.6	30.6	25.8	32.7	42.9	17.2	16.7	19.2	26.3	36.7
	R	97.4	89.7	68.9	47.4	85.3	83.9	46.7	62.9	65.7	66.7	74.5	57.9	63.3
-	P	38.9	78.3	64.0	72.0	75.6	73.7	72.5	81.8	50.8	52.1	48.5	53.8	82.3
	F1	55.6	81.5	66.4	57.2	80.1	78.5	56.8	71.1	57.3	58.5	58.8	55.8	71.5
Dataset		**Yelp DEV**												**TEST**
Model		0	1	2	3	4	5	6	7	8	9	10	11	7
	R	02.1	86.3	85.9	74.0	86.5	74.0	86.1	88.7	80.9	81.0	78.3	82.4	91.0
+	P	38.4	73.5	87.4	87.8	85.0	88.2	86.0	83.8	77.9	78.4	81.9	80.9	86.8
	F1	04.0	79.4	86.7	80.3	86.8	80.5	86.0	86.2	79.4	79.7	80.1	81.6	88.9
	R	00.0	02.5	62.3	80.5	44.0	73.5	53.6	52.9	29.2	30.5	31.3	47.9	55.6
o	P	00.0	40.7	57.0	36.3	57.5	42.6	56.4	55.4	48.5	47.4	49.8	50.4	56.4
	F1	00.0	04.7	59.5	50.5	49.8	53.9	55.0	54.1	36.5	37.1	38.4	49.1	56.0
	R	98.2	93.4	86.0	50.7	91.0	69.3	88.0	84.7	84.0	83.3	88.1	82.9	84.9
-	P	41.0	71.1	88.8	93.5	82.4	88.7	86.0	87.5	72.7	73.0	71.7	82.3	88.8
	F1	57.8	80.7	87.4	65.8	86.5	77.8	87.0	86.1	77.9	77.8	79.1	82.6	86.8
Dataset		**Amazon DEV**												**TEST**
Model		0	1	2	3	4	5	6	7	8	9	10	11	7
	R	06.5	78.7	75.4	93.5	84.1	90.2	91.4	90.5	76.2	77.3	69.9	69.8	91.2
+	P	30.9	80.3	86.7	88.9	85.6	89.6	88.5	88.0	71.5	72.1	80.0	73.9	88.5
	F1	10.7	79.5	80.7	91.1	84.8	89.9	89.9	89.3	73.8	74.6	74.6	71.8	89.9
	R	00.0	09.0	54.0	76.0	33.1	68.1	68.8	70.7	21.0	21.5	25.5	37.5	70.3
o	P	00.0	44.2	64.1	76.2	69.6	78.1	77.4	73.8	74.2	73.7	66.9	59.9	74.4
	F1	00.0	14.9	58.6	76.1	44.8	72.7	72.9	72.2	32.7	33.3	36.9	46.2	72.3
	R	93.7	94.0	88.1	79.7	93.7	86.8	84.7	81.4	86.4	85.2	90.6	83.8	81.5
-	P	33.6	51.7	68.5	83.9	60.7	77.5	78.5	80.3	52.3	52.1	51.9	58.6	79.5
	F1	49.4	66.7	77.1	81.8	73.7	81.9	81.5	80.9	65.2	64.7	66.0	69.0	80.5

Table 3: The Recall (%), Precision (%), and F1 (%) score per category ('+' for positive sentiment, 'o' for neutral, and '-' for negative) across all three datasets. The models are referred to by their model number in table 1, 2. The last column is the results on the test set obtained by our best model, model 7 in table 1.

As such, our results show that ELECTRA does not seem to mesh well with further pre-training, at least at a low number of steps and for its smallest variant. One possible explanation for ELECTRA's sub-par performance when further pre-trained could be that the intricate relationship between its generator and discriminator is being disrupted. During further pre-training, the generator must learn to adjust its distribution over words for the *[MASK]* token to be more oriented towards sentiment-laden words to be successful with the Yelp review data we are using. The discriminator must then adapt to understand that these sentiment-laden words are not replacements, but the originals. When pre-training at this low number of steps, the generator likely does not have enough time to adapt its distribution to place more weight on sentiment-heavy words, and therefore is likely passing the wrong words to the discriminator the majority of the time. This behavior may actually worsen the discriminator's ability to detect replacements. Because the generator is making mistakes far more frequently than it was when it was last checkpointed, a "replaced" token is passed from the generator to the discriminator far more often. In this comparatively easier situation, the discriminator may be learning that increasingly predicting "replaced" is the easiest path to minimizing its own loss in the face of its weakened partner. This process may cause the discriminator to forget many of the learnings about properly detecting replacements it gained from its initial pre-training and actually worsen its performance, which is reflected in its results on the downstream sentiment analysis task. BERT, on the other hand, does not have this lockstep pre-training and can just adjust its mask token prediction distribution to be more sentiment oriented, hence its increasing performance. More experimentation would be needed to see if further pre-training ELECTRA for more steps can get it out of this degenerative state or if it is unrecoverable.

6.3 Joint Fine-tuning Analysis

We will analyze the results of different combinations of training data one by one.

- SST + Yelp: After fine-tuning vanilla ELECTRA on these two datasets, we observed that

the model's performance on SST and Yelp was a bit lower than those exclusively fine-tuned on Yelp or SST, but its overall performance across all the datasets improved. Furthermore, the F1 score on the Amazon dev set is lower for this setting than only fine-tuning on Yelp, indicating that swapping Yelp samples for those from SST in the training data hurts the model's performance on the Amazon dataset. This suggests there is a similarity between the Yelp and Amazon data that is not captured in SST examples.

- Yelp + Amazon: We observe that the performance of this model on the Yelp and Amazon dev sets is worse than the model that fine-tuned on Yelp or Amazon alone, but only slightly. Its performance on Yelp is better than the model that fine-tuned on SST and Yelp, further emphasizing there is some underlying similarity between Amazon and Yelp data that is not shared with SST. In fact, this model performed worse on SST than either model solely fine-tuned on Yelp or Amazon, indicating these two datasets seem to reinforce each other in a way that has a reductive effect on its performance on SST.

- SST + Amazon: Similar to Yelp + Amazon, this combination of training data leads to a performance drop on the SST and Amazon dev sets relative to SST or Amazon only. However, it does better on Yelp than SST or Amazon alone. One reason for this additive benefit could be that the very different SST data makes the model more robust to the slight variations between the similar Amazon and Yelp data. The Amazon samples then provides the model with data similar enough to the Yelp domain to allow it to perform well on Yelp.

- SST + Yelp + Amazon: We can see that this model's performance on SST is better than the one using SST+Yelp or SST+Amazon. However, its performance on the Amazon dataset is worse than any other training data combination that involves the Amazon dataset, but insignificantly so. Although this model's performance on each dataset was not as high as the models trained exclusively on those datasets, it was not very far off (< 2.4 for all three). This is fairly remarkable considering each domain is represented in this model's dataset with only $1/3$ the data that was present for it in the solo experiments. Moreover, this combination gives us the highest Macro-F1 score (73.6) averaged across all datasets out of all the models we have experimented with. Its performance also remained consistent when evaluated on our test set.

Based on the results of our joint fine-tuning experiments, we found that the Yelp and Amazon datasets seem to have some commonalities and can inconsistently lead to gains in the other's performance if one is absent from the fine-tuning. However, the SST dataset is very different than the others and some examples from it must be present in the training set for the model to do well in this domain. One possible explanation is that both the Yelp and Amazon datasets contain user-written reviews from multiple diverse categories (the Yelp dataset has reviews on restaurants, auto shops, etc. The Amazon dataset contains reviews on kitchen supplies, DVD, etc), while the SST dataset is far more homogenous and only contains movie reviews written primarily by critics. Regardless of the similarities or differences between the domains, we found models fine-tuned on only one of our datasets are not directly transferable to another dataset without a significant performance drop, possibly due to semantic clashes between the domains. However, given that the model fine-tuned on a fixed amount of data from all three datasets gave us the best overall performance, we conclude that even small amounts of data from each domain can result in a significant boost in performance for ELECTRA over not having any data from that domain. Moreover, having training examples from differing, but similar domains (Amazon/Yelp) can help ELECTRA with an extremely different domain (SST), as seen by the gains in SST performance of the model fine-tuned using all 3 datasets over those fine-tuned with just 2 (including SST). This finding suggests fine-tuning ELECTRA to achieve strong (though perhaps not peak) performance on sentiment analysis across multiple domains can be accomplished by using just a small amount of training examples from across each of the desired domains.

6.4 Qualitative Analysis

To gain further insight into the performance of Model 7, our best model, we perform attribution analysis with respect to the predicted sentiment for several examples. For these examples, words high-

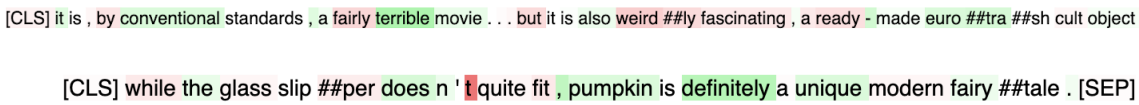

Figure 3: Neutral example from SST misclassified as negative (top) and positive (bottom)

True Label	Predicted Label	Word Importance
2	2 (0.40)	[CLS] the server was really fast and our food came out hot [SEP]
0	2 (0.73)	[CLS] the laptop runs out of battery fast and runs hot [SEP]
0	0 (0.46)	[CLS] the bi ##za ##are dialogue was so fast that it left my ears feeling hot [SEP]

Figure 4: Inputs where the meaning of "hot" and "fast" differ by domain, but the model treats them the same

lighted in green contributed to the probability of the predicted (possibly incorrect) class, while those in red moved the model away from the prediction.

6.4.1 Neutral Examples

Model 7 performed the worst on neutral examples, particularly neutral examples from the SST dataset. 3 shows attribution analysis for a few such misclassified neutral examples (incorrectly classified as positive or negative). As shown here, SST's neutral examples tend to be challenging, even for humans, to decipher as being truly neutral, often easily passing as positive or negative. Moreover, these neutral examples still tend to use fairly charged words, like "terrible", "fascinating", and "unique", each of which tends to strongly move the model towards a prediction of positive or negative rather than neutral. This itself speaks to another finding: Model 7 still tends to give precedence to individual words and their charges over the overall sentence structure when making predictions. For instance, if we change "terrible" in the top example in 3 to the less charged "bad", the model switches the prediction from negative to neutral. As such, the model clearly can detect when an example is neutral, but it can be easily thrown off by particular words.

6.4.2 Cross-Domain Meaning Clash

One area of concern when building a model for cross-domain sentiment analysis is how the model handles words that have different polarity in different domains. Two such words are "fast" and "hot": both tend to have a positive meaning in restaurant reviews (fast service or hot food), but for product reviews and movie reviews often have negative sentiment (running out of battery fast, getting too hot, too fast to understand, etc.). An ideal model should be contextually aware and adapt the effects of those words accordingly. However, as seen in 4, Model 7 does not do this. For the top review (coming from a restaurant domain), both words help the model correctly classify the review as positive, with both having positive attribution for the correct prediction. But the model does not adapt its understanding of those words for the two remaining reviews: it incorrectly views the words as positive in the middle product review, resulting in an incorrect prediction of this negative review as positive; for the bottom movie review, it continues to incorrectly view the words as positive and has them pull away from the correct prediction.

7 Conclusion and Future Work

We studied the effects of further pre-training and fine-tuning ELECTRA for cross-domain sentiment classification. The architectural differences between ELECTRA and BERT make further pre-training ELECTRA more challenging. We found that further pre-training ELECTRA for a small number of steps hurt the model's performance and even small amounts of data from each domain during fine-tuning could boost the model's overall performance across the domains.

Since ELECTRA outperforms BERT even without further pre-training, one way to extend this work is to design a pre-training mechanism that tailors to the lockstep training in ELECTRA. Specifically, we will analyze the individual loss of the generator and discriminator and see if the discriminator outperforms the generator by a large margin. We will also freeze the discriminator every few training steps to give the generator more learning opportunities so that they are on par with each other. Additionally, our experiments on cross-domain fine-tuning can be replicated across a much larger number of domains to see if our findings scale.

8 Authorship Statement

Anfal ELECTRA pre-training, main contributor to ELECTRA fine-tuning, and paper writing.

Meredith Collect and filter Yelp dataset, ELECTRA fine-tuning, and paper writing.

Silvia Collect and filter Amazon dataset, BERT pre-training, ELECTRA fine-tuning, and paper writing.

9 Appendix

Fine-tuning ELECTRA code is based on https://github.com/cgpotts/cs224u/blob/master/hw_sentiment.ipynb

The ELECTRA and BERT repos used for further pre-training are accessible here: https://github.com/google-research/electra and https://github.com/google-research/bert

References

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020a. Adversarial and domain-aware BERT for cross-domain sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4019–4028, Online. Association for Computational Linguistics.

Yongping Du, Meng He, Lulin Wang, and Haitong Zhang. 2020b. Wasserstein based transfer network for cross-domain sentiment classification. *Knowledge-Based Systems*, 204:106162.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Christopher Potts, Zhengxuan Wu, Atticus Geiger, and Douwe Kiela. 2020. Dynasent: A dynamic benchmark for sentiment analysis. *CoRR*, abs/2012.15349.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2020. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4933–4941, Marseille, France. European Language Resources Association.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020. SentiX: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 568–579, Barcelona, Spain (Online). International Committee on Computational Linguistics.