Investigating Numerical Reasoning in BERT-Based Models

Derek Driggs

Collaborators: Stephen Chestnut, Przemysław Kaleta, Jamie Brunning, Evgeni Kirov, and Andrew Smith

Abstract

Pre-trained language models are able to encode enormous amounts of linguistic information, but these models fail to capture information critical to numerical reasoning. Even though these language models have achieved state-of-the-art performance in a variety of reading comprehension tasks, they often fail to perform simple tasks requiring numeracy, severely limiting their performance in financial applications.

In this work, we develop tests to evaluate the numeracy of pre-trained language models and show that BERT-based models fail to perform simple numerical reasoning tasks. We also introduce NumBERT, a BERT-based model with significantly improved numerical reasoning abilities. The smallest model we develop, NumBERT-base, achieves 73.6 F1 on DROP, outperforming all existing models of similar size. Our large model, NumBERT-large, achieves 80.4 F1 on DROP, compared to a state-of-the-art of 84.8 F1 for a model of similar size.

1. Introduction

Developing reading comprehension models with numerical reasoning skills has recently become an important focus of the NLP community, and this topic is particularly relevant for researchers interested in financial applications. Much of this research was sparked by the release of the Discrete Reasoning Over Paragraphs (DROP) dataset cite, a question-answering dataset whose questions require numerical reasoning to answer. This dataset provides researchers with a benchmark to test the relative numeracy of their models, and it influenced the development of several new reading comprehension models with improved numeracy.

This work makes three contributions toward this line of research:

1. We investigate BERT's numerical reasoning when performing masked language modelling tasks and the emergence of numeracy in its attention patterns;
2. We introduce NumBERT, a BERT-based model developed to capture numerical information in its embeddings that outperforms BERT on a variety of tasks requiring numerical reasoning;
3. We show that our smaller model NumBERT-base significantly outperforms models of similar size on the DROP question-answering task, and NumBERT-large performs competitively.

Figure 1. BERT's predictions for masked numeral prediction tasks requiring numerical comparisons. Although BERT appears to perform well in the experiment on the left, it usually parrots the first token of the sequence.

2. Evaluating Numeracy in BERT

To see how numeracy emerges in BERT, we analyse its performance on a masked numeral prediction task and observe its attention patterns while performing this task. We find that BERT is able to encode numerical relationships between words and numbers, but it struggles to perform numerical comparisons, and it "cheats" when it can.

2.1. Numeric Comparisons

We test the BERT encoder to complete the sentence "X is greater than [MASK]", where we vary X from 0 to 100, and we represent X as a word, integer, and float ("one", "1", and "1.0", respectively). To exhibit numeracy, BERT should predict a numeral for the [MASK] token so that the resulting sentence is true. BERT should also make similar predictions when the sentence is changed to "[MASK] is greater than X", or "X is less than [MASK]".

Figure 2. Visualising attention in BERT. Thicker lines indicate that the word on the left attends strongly to the word on the right.

Figure 1 shows the results of these experiments. We do not include a plot for the sentence "[MASK] is greater than X" because the model predicted "it" (a non-numeric token) in over 95% of the examples. We notice several patterns:

1. BERT is more likely to predict a numeral when "[MASK]" appears at the end of the sentence.
2. When BERT predicts a numeral from a float, it often predicts the integer part (i.e., "88.0 is greater than [88]").
3. When BERT predicts a numeral from a word, it predicts the greatest multiple of 10 that is less than X (i.e. "eighty-eight is greater than [eighty]").

From these observations, we can conclude that BERT does not use numerical reasoning to infer a reasonable answer, but it "cheats" using surrounding information to extract a guess. When the numeral appears at the end of the sentence, BERT produces the sentence "[it] is greater than X", likely because "it" is a common subject. This explains pattern (1). When the numeral appears at the beginning, BERT often uses the first numeral it sees to fill the [MASK] token. Numerals represented as floats and words are split into multiple tokens, explaining the patterns (2) and (3).

This experiment demonstrates the importance of evaluating BERT-based models carefully when using them for new applications, as they can often "fake" strong performance. This supports recent findings of author et al. (Beyond accuracy), where the authors demonstrate that BERT's strong performance is often fragile. The authors show that small changes, such as adding an extra period to the end of a sentence, can cause BERT to drastically alter its behaviour.

2.2. Attention Patterns

A popular heuristic to understand BERT's behaviour is to look at its attention patterns. BERT-base contains 12 attention layers, each containing 12 attentions heads. On a high level, each of these heads take in the word embeddings from the previous attention layer and forms output embeddings by taking weighted averages of the inputs. Precisely, for a sentence containing n words and a layer with hidden dimension d_h, the output embeddings $H_{\text{out}} \in \mathbb{R}^{n \times d_h}$ are computed from the input embeddings $H_{\text{in}} \in \mathbb{R}^{n \times d_h}$ as

$$S = \text{Softmax}\left(\frac{H_{\text{in}} W_1 (H_{\text{in}} W_2)^\top}{\sqrt{d_h}}\right) \qquad (2.1)$$
$$H_{\text{out}} = S H_{\text{in}} W_3,$$

where $W_1, W_2, W_3 \in \mathbb{R}^{h \times h}$ are parameter matrices. Heuristically, the weights in S determine how much each input embedding influences each output embedding. If $S_{i,j}$ is close to 1, then the output embedding of the i^{th} word depends strongly on the input embedding of the j^{th} word.

Figure 3. Showing the number of attention heads in BERT where the [MASK] mostly attends to the numeral and vice-versa. Self-attention and attention to [CLS] and [SEP] are ignored.

In the experiment of the previous section, we use attention patterns to see where BERT attends when it is making its prediction. We would expect a model with numerical reasoning abilities to attend mostly to the unmasked numeral, and a model making a random prediction would demonstrate no coherent attention pattern.

Figure 3 shows the attention patterns in every head of BERT for the sentence "five is greater than four".[1] Head 11 in Layer 1 is highlighted, showing strong attention between "five" and "four". A word attending to [CLS], [SEP], or itself is common and considered a "no-option" stance CITE what does BERT look at? Clark et al.. Ignoring these types of attention, "five" attends mostly to "four", and vice-versa.

Performing this test systematically, we consider the sentence "X is greater than [MASK]" from the experiment of the previous section, and analyse attention patterns using the sentence completed by BERT. Figure 2 shows the number of heads where X attends mostly to [MASK], and vice-versa. When X is represented as a word or a float, it is split into several tokens, so for a fair comparison we include a plot measuring attention for only the first token and one measuring the sum of the attentions over all tokens representing X.

[1] The [CLS] and [SEP] tokens are included to denote the beginning and end of a sentence, respectively.

When measuring only the first token, this attention pattern is strongest when X is represented as an integer. However, when the attention to the tokens "." and "0" are included for the float, then this attention pattern appears to be the strongest for float values. This is counter-intuitive because all the relevant information for the float is included in the first token, so it is unexpected for BERT to attend to the entire float.

Overall, these experiments suggest that BERT assigns large attention weights among numerals when performing a task requiring numerical reasoning. This hints at an understanding that numerals differ from non-numeric words

2.3. Predicting Numerals from Words

While the previous experiment evaluates BERT's understanding of numerical relationships, the experiment of this sections measures how well BERT understands relationships between numbers and words. For our next experiment, we consider the sentence "[Organization] [quantity] [verb] [MASK] %.", and record the predicted numeral. We consider 5 organisations (e.g. Tesco, Pacificare Health), 4 quantities (e.g. profits, revenues), and 12 verbs (e.g. dip, rise, soar). We test the following hypotheses:

- Changing the company name should not affect the predicted numeral.
- The sequence of verbs "dip, rise, soar" should produce a sequence of non-decreasing numerals.
- The verbs "slightly fall" and "slightly rise" should produce strictly smaller numerals than "significantly fall" and "significantly rise", respectively.

Table REF shows the results of this experiment. BERT performs surprisingly well representing the "magnitude" of a verb in its prediction, but is struggles with more complex examples. Qualifying adverbs, such as "slightly" and "significantly" do not appear to affect BERT's predictions.

These experiments complement the recent work of spying, who show that BERT embeddings, more than other contextualized embeddings, distribute identifying features among the embeddings of multiple words in a sentence. In this experiment, the magnitude of the predicted numeral is represented not only in its own embedding, but also in the embedding of the verb, so varying only the verb of the sentence can reliably change the predicted numeral.

These experiments show that a certain degree of numeracy emerges within attention layers to perform masked language modeling tasks, but also that BERT's numeracy is limited. In the sequel, we discuss two recently proposed methods to improve numeracy in language models and apply them to BERT.

Operation	Example
arithmetic	23.5 - 432 + 3 - 5431.56
min/max	largest(42, 12.4, 53.78)
date min/max	oldest(June 04, 1959; 31 May, 13)

Table 1. Examples from the synthetic numeric dataset developed to pre-train GenBERT cite.

3. NumBERT

Most recent works improve BERT's performance on numerical reasoning tasks by appending "calculator" output layers to perform counting and arithmetic cite drop, mtmsn, bert calc. While these additions improve performance for specific tasks such as question-answering on DROP, they do improve numeracy in BERT's embeddings, but outsource numerical reasoning to the external calculators, making these approaches difficult to generalise to other applications. We focus on two approaches that improve numeracy in the embeddings themselves.

3.1. NumNet

Authors et al. recently introduced NumNet, a QANet-based architecture cite that improves numeracy in the embeddings by constructing numbers' ordinal relationships into the network's architecture. On top of the QANet encoder, NumNet inputs the embeddings of the numerals into a numerically-aware graph neural network (NumGNN) and replaces them with the network's output.

The NumGNN is represented by a directed graph $\mathcal{G} = (\mathbf{V}; \mathbf{E})$ where the nodes $\mathbf{V}$ are the numbers in the input sentence and the edges $\mathbf{E}$ encode the numbers' ordinal relationships. An edge $e_{ij} \in \mathbf{E}$ connects node v_i to node v_j if $v_i \geq v_j$. Let h^i_{in} be the encoder embedding of the numeral v_i. The NumGNN updates this embedding using the following procedure:

$$\begin{aligned} \alpha_i &= \text{Sigmoid}(W_v h^i_{\text{in}} + b_v), \\ \tilde{h}^i &= \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \alpha_j W_{r_{ij}} h^j_{\text{in}}, \\ h^i_{\text{out}} &= \text{ReLU}(W_f h^i_{\text{in}} + \tilde{h}^i + b_f). \end{aligned} \tag{3.1}$$

Here, α_i measures the relevance of the numeral v_i to numeral v_j, so that unrelated numbers do not affect each others' embeddings. The set $\mathcal{N}_i = \{j : (v_j, v_i) \in \mathbf{E}\}$ contains the neighbors of v_i. The index $r_{ij} \in \{<, \geq\}$ is determined by the ordinal relationship of the numbers (so $r_{ij} \in \{\geq\}$ if $e_{ij} \in \mathbf{E}$ and $r_{ij} \in \{<\}$ otherwise). The matrices $W_v, W_f, W_{\geq}$, and $W_<$ are parameter matrices, and b_v and b_f are biases.

In NumBERT, we use this NumGNN to update the numeral

P: The commander recruited 1949 Polish families in Spain. The householder recruited 1996 Japanese families in Spain. There were 10913 rebels and 77 Chinese families in Spain. 6641 British soldiers, 476 asian rebels, and 338 Germans families were recruited in Russia.
Q: How many Japanese families were in Spain? **A:** 1996

Table 2. Examples from the synthetic textual dataset developed to pre-train GenBERT cite.

embeddings of BERT's final layer. As we show in Section 4, these embeddings capture significantly more numerical information than those produced using only the BERT encoder.

Although incorporating the NumGNN of NumNet into BERT's architecture improves its numerical reasoning abilities, it only updates the final-layer embeddings.[2] The NumGNN cannot restore numerical information that might have been lost in the earlier layers of BERT. To optimise performance, we initalise the NumBERT encoder with weights pre-trained to perform numerical reasoning tasks.

3.2. GenBERT

In order to circumvent using external calculators to produce numeric answers for the DROP task, GenBERT cite attaches a decoder to the BERT-base encoder and trains the model to decode numeric answers. The decoder weights are tied to BERT's encoder, so GenBERT has roughly the same number of parameters as BERT-base, and fewer parameters than BERT-based models relying on external calculators. Also, because the weights are tied, the numeracy GenBERT learns during training is represented in the weights of the BERT encoder, and these weights can then be used to initialise fine-tuning for other applications.

Importantly, GenBERT uses *digit tokenization*, which greatly improves its numeracy. While BERT's sub-word tokenization often splits numbers into unpredictable components, for example, $339 \rightarrow 3, \#\#39$, GenBERT splits numerals systematically; $339 \rightarrow 3, \#\#3, \#\#9$. Number representations are also standardised, so words are translated to numerals, and extra characters, such as "," in "1,000", are removed. As the authors show, this approach to tokenization is crucial for improved numerical performance cite paper, section.

For training data, the authors generate synthetic numeric data and numerically-focused textual data. The numeric data contains examples of mathematical operations, such as arithmetic expressions and argmax computations. The textual data comprises simple passages involving numbers of objects, and questions requiring arithmetic manipulation of these numbers. Examples from both synthetic datasets are included in Table 2. In addition to these synthetic datasets, the authors also use English Wikipedia articles on topics involving numbers and dates (for example, economics and history, see paper, Appendix A.3 for details).

To train GenBERT, the authors minimise the sum of two objectives. The first is the standard masked language model (MLM) objective used to train the original BERT model cite:

$$\mathcal{L}_{\text{mlm}}(\langle \mathbf{m} \rangle) = -\frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \log(p(a_i | i, \langle \mathbf{m} \rangle)). \quad (3.2)$$

Here, $\langle \mathbf{m} \rangle$ is a sequence of tokens containing some masked words, $\mathcal{M}$ is the index set of the masked tokens, and a_i are the ground-truth for the masked tokens. The probability $p(a_i | i, \langle \mathbf{m} \rangle)$ is computed by passing the output of the BERT encoder through a feed-forward layer just as it is during BERT's pre-trianing. The MLM objective is minimised over the Wikipedia dataset only.

The synthetic data is used to train GenBERT's decoder. In this task, the encoded inputs H_{enc} and answer sequence $\langle \mathbf{a} \rangle = (a_0, \cdots, a_{m+1})$ are input into the decoder. Using H_{enc} and the answer tokens up to index i, the decoder predicts $p_{\text{dec}}(a_{i+1} | a_0, \cdots, a_i, \langle \mathbf{m} \rangle)$, so the probability of producing the full answer $\langle \mathbf{a} \rangle$ is

$$p_{\text{dec}}(\langle \mathbf{a} \rangle | \langle \mathbf{m} \rangle) = \prod_{i=0}^{m} p_{\text{dec}}(a_{i+1} | a_0, \cdots, a_i, \langle \mathbf{m} \rangle). \quad (3.3)$$

This objective is minimised over the synthetic datasets only, and is the source of GenBERT's improved numeracy over BERT.

Our proposed model, NumBERT, uses GenBERT's pre-trained weights for initialisation, and trains the NumGNN during task-specific fine-tuning.

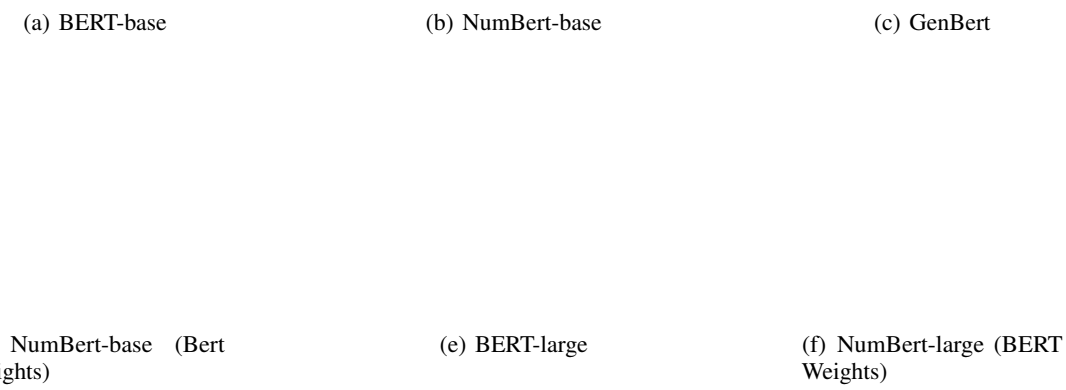
4. Numerical Experiments

We compare BERT-base, NumBERT-base, BERT-large, and NumBERT-large on three numerical tasks:

1. Decoding: regressing a numeral to its magnitude,
2. Sentiment analysis on financial news headlines, and
3. Question-answering on the DROP dataset.

We include ablation test for NumBERT as well, comparing the full NumBERT models to NumBERT without using the GenBERT weights, and GenBERT without using a NumGNN. Due to the time constraints of this project, we

[2] A natural solution is to attach NumGNNs to all of the encoder layers, but this does not seem to improve performance. See Section 4.4 for information on these experiments.



(a) BERT-base (b) NumBert-base (c) GenBert

(d) NumBert-base (Bert Weights) (e) BERT-large (f) NumBert-large (BERT Weights)

Figure 4. Predictions on the test set of the decoding task. The models were trained on the interval $[-500, 500]$ to regress from a numeral to its value.

Model	MSE on Test Set
BERT-base	2013.5
NumBERT-base	**450.1**
GenBERT	594.3
NumBERT-base (BERT weights)	1780.5
BERT-large	2301.6
NumBERT-large (BERT weights)	**637.9**

Table 3. Decoding test results showing the MSE over the test set of models trained to regress from a numeral to its value.

are unable to complete the GenBERT pre-training task using BERT-large, so NumBERT-large is initialised using the BERT-large weights. Pre-training larger GenBERT models is in progress, and we believe this is a promising direction for future research.

4.1. Decoding

The decoding task tests how well a numeral's magnitude is represented in its embedding. To test a model, we train a three-layer fully connected network with ReLU activations on top of the encoder to regress from a number's embedding to its value. We use mean-square error (MSE) loss. For datasets, we randomly select 80% of the integers in the interval $[-500, 500]$ to use for training and use the remaining 20% for testing. We evaluate performance on the interval $[-1000, 1000]$ to observe how well models extrapolate outside the region $[-500, 500]$.

Because the NumGNN in NumBERT must be pre-trained, we initialise it with weights obtained through fine-tuning on the DROP dataset (see Section 4.3), and we freeze these weights during training. We also append the numbers -500 and 500 to the input sequences of NumBERT so that the NumGNN has numeral embeddings available to perform numerical comparisons. Digit tokenization is used for all models incorporating GenBERT weights. Each model is trained for 1000 epochs using a batch size of 128 and a step size of 3×10^{-5} that decreases by a factor of 0.95 every 10 epochs.

Figure 5 and Table 3 display the results of this test. Surprisingly, the smaller models perform better on this task than the larger models, even though the number of parameters updated during training is the same for all models. For both sizes, NumBERT demonstrates significant improvement over BERT. The results suggest that much of this improvement comes from initialising with the GenBERT weights, but incorporating the NumGNN on top of BERT's weights and GenBERT's weights leads to further improvement in both cases. All models extrapolate poorly.

4.2. Sentiment Analysis

A quintessential application of NLP models in finance is sentiment analysis, and the sentiment of financial news

Model	Acc. Total	Acc. Numeric
BERT-base	59.0	52.3
NumBERT-base	59.2	**62.9**
GenBERT	59.0	58.9
NumBERT-base (BERT weights)	**60.6**	58.4
BERT-large	**61.7**	**60.9**
NumBERT-large (BERT weights)	61.5	59.6

Table 4. Accuracy on sentiment analysis task. Accuracy is calculated on the entire test set and on test examples requiring numeracy.

often depends on numerical information. Our dataset of financial news headlines contains 2610 financial news headlines with labels that are distributions over the classes "Unknown","neutral", "strongly negative", "negative", "weakly negative", and analogous labels for positive sentiment. Labels also included whether numeracy is required to extract sentiment from the feature. look up train/dev split.

For training, we append a linear layer onto each model and minimise the KL-divergence between the predicted and ground-truth distributions. In contrast to the decoding task, the model weights are updated during training. Because we are fine-tuning, we do not initialise the NumGNN of the NumBERT models with pre-trained weights. We append the numbers 0,5,10,20, and 100 to the end of each input sequence when training a NumBERT model so the NumGNN always has numeral embeddings to use for comparisons. Digit tokenization is used for all models employing GenBERT weights. Each model is trained for 15 epochs using a step size of 3×10^{-5} for the model parameters, and a step size of 10^{-3} for the classifier layer parameters. A batch size of 32 was used for the base models and 16 for the large models.

Table 4 shows the results of this experiment. The results suggest that all models exhibit roughly the same performance, with larger models performing slightly better overall. Because results can vary significantly over many runs, future work should aggregate the performance of these models over several experiments for a more accurate representation of their performance.

To test whether the models learned how numbers should effect the sentiment of a headline, we used the trained models to assign a sentiment score to the headline "Pacificate Health Q2 revenues up X percent from one year ago", where X is varied from 0 to 100. This example, taken from the test set, stress-tests the models' numeracy. The models should assign higher probabilities to positive sentiment values as X increase, and the models should ignore the "red-herring" numbers in "Q2" and "one year". Figure **??** at the end of this report shows the effect of revenue increase on the sentiment score from each model, with NumBERT and its components clearly outperforming BERT.

4.3. Question-Answering on DROP

Finally, we evaluate our models on the DROP question-answering task. The DROP dataset contains passages, questions that can be answered from the associated passage, and answers to the questions. The questions are designed to require numerical reasoning to answer. The answers can be a span or multiple spans from the passage, or numbers and dates not appearing in the passage. Figure 6 shows some examples. This dataset is a standard benchmark for evaluating numeracy in NLP models. Many models, including NumNet and GenBERT, were developed specifically to push the state-of-the-art performance on DROP.

To fine-tune our models for question-answering on DROP, we add four standard output layers onto their encoders. Two of these are passage-span and question-span extraction layers. These layers predict the starting index and stopping index for an answer span in the passage and question, respectively. Because these are standard output layers for question-answering model, we refer to cite bert, section for further details.

Many works (including the experiments of this section) have shown that adding counting and arithmetic expression output layers dramatically improves performance on DROP cite the many works. Counting layers predict an output from 0 to 9 and are useful for producing numerical answers that are not contained in the passage or question. Arithmetic expression layers predict a sign "$-$" or "$+$" for each of the numbers in the passage and question, and output a string of signed numerals to show how to produce a numerical answer. These output layers are also standard, so we refer to cite DROP for a detailed discussion.

Table 5 shows the results. NumBERT performs significantly better than BERT-based models of similar sizes. For NumBERT-base, including the NumGNN without using the GenBERT weights leads to significant performance improvement, and GenBERT demonstrates enormous improvement over BERT as well. Importantly, the improvements offered by the GenBERT weights and the NumGNN are somewhat orthogonal, so combining the two approaches leads to further performance improvements.

4.4. Additional Numerical Experiments

During this project, we trained and tested several other models that did not exhibit improved performance, but still provide useful information. Because the NumBERT GNN only adjusts the numeral embeddings of the final layer, we

Model	EM	F1
BERT-base (span only)	27.9	31.8
BERT-base	54.57	57.64
NumNet	64.6	68.0
NumBERT-base	**69.3**	**73.2**
GenBERT	68.3	70.3
NumBERT-base (BERT weights)	67.6	71.4
BERT-large	70.6	74.3
NumBERT-large (BERT weights)	**76.7**	**80.3**
NumNet+v2	81.5	84.8

Table 5. Exact-match (EM) accuracies and F1 scores on the DROP development set. "Span only" indicates a model without the counting and arithmetic output layers. NumNet+v2 is a BERT-large sized model initialised using RoBERTa-large and including several other improvements over NumNet.

trained a NumBERT-base model that incorporates multi-layer GNNs, hoping to capture more numerical information in the embeddings of every layer. We added a GNN to the final n layers of BERT-base with $n \in \{1, 2, 6, 12\}$, and every model performed roughly the same on the DROP question-answering task (but the training time increased considerably with the addition of each layer).

We also attempted to perform the GenBERT pre-training procedure on BERT-large and ALBERT-xxlarge cite. Both models are still slowly converging, but training is far from complete. Using 8 GPUs, I estimate that it will take 2-4 weeks for either of the models to complete training, although ALBERT-xxlarge appears to be converging much faster than BERT-large. Scaling the GenBERT pre-training procedure to larger models is a promising direction for future research, and we will continue to investigate this after the internship ends.

5. Conclusion

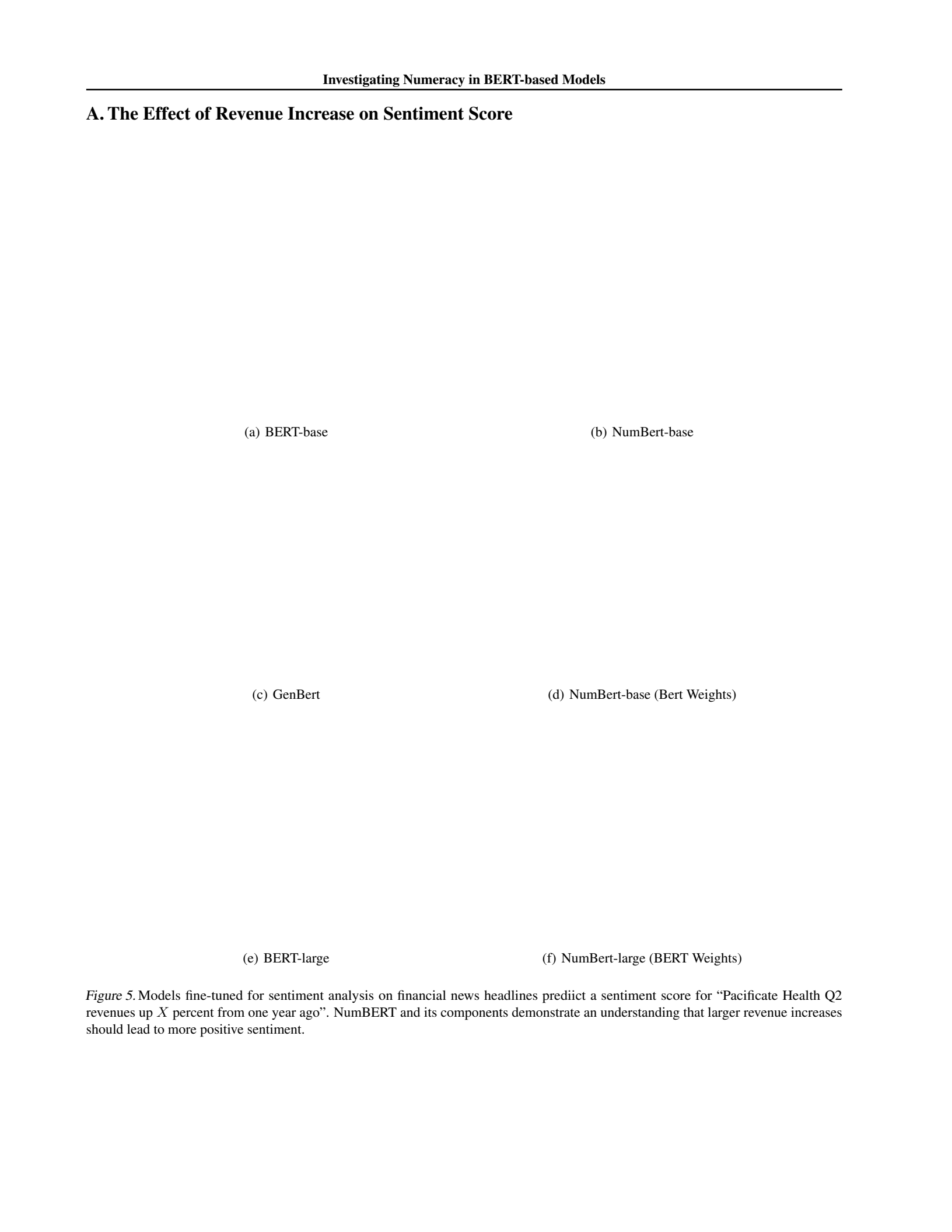
In this project, we investigated how numeracy emerges in BERT-based models and developed NumBERT, a BERT-based model with improved numerical reasoning abilities. We found that BERT demonstrates some numerical reasoning abilities when performing the masked language modeling tasks, but regularly fails and cheats. When performing numerical comparisons to complete the sentence "X is greater than [MASK]", BERT often repeats the first numeral in the sequence without demonstrating any numerical reasoning. However, its attention patterns show a clear relationship between the numeral and the masked token, suggesting a relationship between BERT's embedding of the two tokens. BERT is also occasionally able to adjust its predicted numeral to match the surrounding context, for example, to match the "magnitude" of a verb describing stock price and revenue increase.

By injecting numeracy into BERT's embeddings, NumBERT outperforms BERT on a variety of numerical reasoning tasks. Our decoding experiments show that NumBERT-base and NumBERT-large encode more information relating to the magnitude of a numeral in their embeddings than do similarly sized BERT models. NumBERT-base significantly outperforms models models of similar size on the DROP dataset. Although NumBERT large does not perform as well as the state-of-the-art model of the same size NumNet+v2, many contributions of NumNet+v2 can be incorporated into NumBERT-large in future works, including initialising using the weights of RoBERTa-large cite rather than BERT-large, and using a simple multi-span extraction technique cite tase.

A promising direction for future research is to scale the pre-training approach of GenBERT to larger models. As demonstrated in our experiments GenBERT pre-training procedure significantly improves the model's numerical reasoning ability, and seeing this level of improvement in large models such as RoBERTa-large or ALBERT-xxlarge could lead to state-of-the-art performance on numerical reasoning tasks.

References

A. The Effect of Revenue Increase on Sentiment Score



(a) BERT-base

(b) NumBert-base

(c) GenBert

(d) NumBert-base (Bert Weights)

(e) BERT-large

(f) NumBert-large (BERT Weights)

Figure 5. Models fine-tuned for sentiment analysis on financial news headlines prediict a sentiment score for "Pacificate Health Q2 revenues up X percent from one year ago". NumBERT and its components demonstrate an understanding that larger revenue increases should lead to more positive sentiment.

B. DROP Dataset Examples

Reasoning	**Passage** (some parts shortened)	**Question**	**Answer**
Subtraction (28.8%)	That year, his **Untitled (1981)**, a painting of a haloed, black-headed man with a bright red skeletal body, depicted amid the artists signature scrawls, was **sold by Robert Lehrman for $16.3 million, well above its $12 million high estimate**.	How many more dollars was the Untitled (1981) painting sold for than the 12 million dollar estimation?	4300000
Comparison (18.2%)	In **1517, the seventeen-year-old King sailed to Castile**. There, his Flemish court .... **In May 1518, Charles traveled to Barcelona in Aragon**.	Where did Charles travel to first, Castile or Barcelona?	Castile
Selection (19.4%)	In 1970, to commemorate the 100th anniversary of the founding of Baldwin City, **Baker University professor and playwright Don Mueller and Phyllis E. Braun, Business Manager, produced a musical play entitled The Ballad Of Black Jack** to tell the story of the events that led up to the battle.	Who was the University professor that helped produce The Ballad Of Black Jack, Ivan Boyd or Don Mueller?	Don Mueller
Addition (11.7%)	Before the UNPROFOR fully deployed, the HV clashed with an armed force of the RSK in the village of Nos Kalik, located in a pink zone near Šibenik, and captured the village at 4:45 p.m. on **2 March 1992**. The JNA formed a battlegroup to counterattack the **next day**.	What date did the JNA form a battlegroup to counterattack after the village of Nos Kalik was captured?	3 March 1992
Count (16.5%) and Sort (11.7%)	Denver would retake the lead with kicker **Matt Prater nailing a 43-yard field goal**, yet Carolina answered as kicker **John Kasay ties the game with a 39-yard field goal**. ... Carolina closed out the half with **Kasay nailing a 44-yard field goal**. ... In the fourth quarter, Carolina sealed the win with **Kasay's 42-yard field goal**.	Which kicker kicked the most field goals?	John Kasay
Coreference Resolution (3.7%)	**James Douglas** was the second son of Sir George Douglas of Pittendreich, and Elizabeth Douglas, daughter David Douglas of Pittendreich. Before **1543 he married Elizabeth**, daughter of James Douglas, 3rd Earl of Morton. **In 1553 James Douglas succeeded to the title and estates of his father-in-law**.	How many years after he married Elizabeth did James Douglas succeed to the title and estates of his father-in-law?	10
Other Arithmetic (3.2%)	Although the movement initially gathered some **60,000 adherents**, the subsequent establishment of the Bulgarian Exarchate **reduced their number by some 75%**.	How many adherents were left after the establishment of the Bulgarian Exarchate?	15000
Set of spans (6.0%)	According to some sources 363 civilians were killed in **Kavadarci**, 230 in **Negotino** and 40 in **Vatasha**.	What were the 3 villages that people were killed in?	Kavadarci, Negotino, Vatasha
Other (6.8%)	This **Annual Financial Report** is our principal financial statement of accountability. The **AFR gives a comprehensive view** of the Department's financial activities ...	What does AFR stand for?	Annual Financial Report

Figure 6. Examples from the DROP dataset, showing passages, questions, answers, and the type of numerical reasoning required to answer each question. This figure was originally published in cite drop.