Towards BERT-based Automatic ICD Coding: Limitations and Opportunities

Damián Pascual Sandro Luck Roger Wattenhofer
ETH Zurich, Switzerland
{dpascual,wattenhofer}@ethz.ch, sluck@student.ethz.ch

Abstract

Automatic ICD coding is the task of assigning codes from the International Classification of Diseases (ICD) to medical notes. These codes describe the state of the patient and have multiple applications, e.g., computer-assisted diagnosis or epidemiological studies. ICD coding is a challenging task due to the complexity and length of medical notes. Unlike the general trend in language processing, no transformer model has been reported to reach high performance on this task. Here, we investigate in detail ICD coding using PubMedBERT, a state-of-the-art transformer model for biomedical language understanding. We find that the difficulty of fine-tuning the model on long pieces of text is the main limitation for BERT-based models on ICD coding. We run extensive experiments and show that despite the gap with current state-of-the-art, pretrained transformers can reach competitive performance using relatively small portions of text. We point at better methods to aggregate information from long texts as the main need for improving BERT-based ICD coding.

1 Introduction

During patient stays in medical institutions, clinicians generate text notes that record the state of the patient as well as the diagnoses and the treatments administered. Typically, a code from the International Classification of Diseases (ICD) is assigned to these clinical notes, in order to provide standardized information about the patient condition. ICD codes are used for different purposes, such as billing, computer-assisted diagnosis or epidemiological studies (Choi et al., 2016; Denny et al., 2010; Avati et al., 2018). Assigning ICD codes to medical notes is usually done manually by clinicians. This is an error-prone and time-consuming procedure and therefore, automatic solutions have been studied for over two decades (Larkey and Croft, 1996; de Lima et al., 1998).

However, automatic ICD code assignment proves challenging for multiple reasons. First, there exists a very large number of ICD codes (17.000) and each clinical report may have associated more than one code. To deal with this large multi-label classification problem, it is common to reduce the number of codes to those that appear most frequently (Mullenbach et al., 2018). Second, medical text usually lacks structure, includes irrelevant passages, as well as abbreviations, misspellings, numbers and a very specific vocabulary. On top of that, medical notes are long, which makes it difficult for automatic coding models to draw relations between different sections of the reports.

Current state-of-the-art methods for automatic ICD coding from medical notes are based on deep learning (Wang et al., 2018b; Mullenbach et al., 2018; Vu et al., 2020). These methods use different configurations of convolutional (CNN) and recurrent (RNN) neural networks as well as attention modules(Bahdanau et al., 2014). This stands in contrast to most areas of natural language processing (NLP), where models based on the transformer architecture (Vaswani et al., 2017) dominate the state-of-the-art (Wang et al., 2019). One of the main strengths of transformer models is their ability to deal with long range dependencies. This is a desirable property in ICD coding, where an understanding of different parts of the document may be necessary to assign a code. The lack of transformer models for ICD coding is surprising, especially since there already exist BERT-based models (Devlin et al., 2019) (a type of bidirectional transformer) that are trained on medical text data (Lee et al., 2020; Alsentzer et al., 2019; Gu et al., 2020). These models have achieved state-of-the-art performance on other tasks such as named entity recognition or question answering on medical documents (Gu et al., 2020).

On the other hand, the complexity of transformers scales quadratically with the length of their in-

Proceedings of the BioNLP 2021 workshop, pages 54–63

put, which restricts the maximum number of words that they can process at once. This limitation may be critical in ICD coding, since clinical notes usually exceed this maximum input length. In this work, we investigate in detail BERT-based ICD coding, and explore different strategies to overcome the constraint on the input length by using an encoder-decoder architecture. We use the MIMIC-III dataset (Johnson et al., 2016), a big and widely used dataset for the ICD coding task, in order that our results are directly comparable to other existing methods (Wang et al., 2018b; Mullenbach et al., 2018; Vu et al., 2020). By exposing the limitations and benefits of BERT-based models on this task our work sets a solid basis for further research on automatic ICD coding systems.

2 Related Work

Automatic ICD coding has been an active area of research for over two decades. Already Larkey and Croft (1996) and de Lima et al. (1998) proposed different strategies to extract features from medical documents in order to build classifiers for automatically assigning ICD codes to medical notes. More recently, Perotte et al. (2014) proposed a multi-level Support Vector Machine (SVM) model to predict ICD codes from the MIMIC-II dataset (Saeed et al., 2011), the precursor of the MIMIC-III dataset (Johnson et al., 2016) that we consider in this work. Similarly, Scheurwegs et al. (2017) presented a method to extract features from structured and unstructured text and evaluated it on the MIMIC-III dataset.

In the last years, the state-of-the-art of automatic ICD coding has been dominated by deep learning models. Shi et al. (2017) proposed an LSTM model that operates at the character-level combined with an attention mechanism (Bahdanau et al., 2014). Wang et al. (2018b) proposed an embedding model based on GloVE embeddings (Pennington et al., 2014) that maps text and labels to the same space, where predictions are made using the cosine similarity. Mullenbach et al. (2018) proposed a model that combined convolutions with a per-label attention mechanism. This model was further improved by Xie et al. (2019) and Li and Yu (2020). Vu et al. (2020), proposed a label-attention model that reached the current best performance for ICD coding on the MIMIC-III dataset. All of these works represent only a portion of the research carried out in this field (Karimi et al., 2017; Baumel et al., 2018; Song et al., 2020; Prakash et al., 2017; Cao et al., 2020).

Since the appearance of the Transformer model (Vaswani et al., 2017), transformer-based architectures (Brown et al., 2020; Lewis et al., 2020; Raffel et al., 2019) have become state-of-the-art in almost every area of Natural Language Processing (Wang et al., 2018a, 2019) thanks to their ability to handle long range dependencies. BERT (Devlin et al., 2019), a bidirectional transformer, is of particular importance since it is the basis of many other language understanding models. Nonetheless, given the specific characteristics of medical text, e.g., specialized vocabulary, models pretrained on generic language, like BERT, do not reach high performance on biomedical language understanding tasks. Therefore, specialized models, such as BioBERT (Lee et al., 2020) or ClinicalBERT (Alsentzer et al., 2019), pretrained on medical text have been proposed. In particular, the recent PubMedBERT model (Gu et al., 2020) is the state-of-the-art in the BLURB benchmark (Gu et al., 2020), a benchmark for biomedical language understanding which includes the following tasks: named entity recognition, question answering, document classification, relation extraction, sentence similarity and evidence-based medical information extraction. Despite its prominence in medical language understanding, automatic ICD coding escapes the set of tasks where BERT-based models excel. To the best of our knowledge, no BERT-based model has been proposed yet that reaches competitive performance on ICD coding on the MIMIC-III dataset. In this work, we investigate in detail BERT-based ICD coding and identify existing limitations and opportunities.

3 Background

In this section we present the BERT model used in our experiments as well as the evaluation metrics.

3.1 PubMedBERT

PubMedBERT (Gu et al., 2020) is a transformer model with the same architecture as BERT-base (Devlin et al., 2019), i.e., it has 12 transformer layers, 100 million parameters and it outputs vector representations of 768 elements. PubMedBERT is trained from scratch on PubMed text, on a dataset of 3.1 billion words (21 GB). Furthermore, Pub-

MedBERT has not been pretrained on the MIMIC datasets as ClinicalBERT (Alsentzer et al., 2019) or BlueBERT (Peng et al., 2019), and therefore, we can evaluate it on MIMIC-III without information leakage from the test set. We choose this model among the existing ones because it is currently the state-of-the-art in biomedical understanding tasks as measured by the BLURB benchmark[1]. We use the implementation from HuggingFace (Wolf et al., 2019).

3.2 Evaluation Metrics

Following previous work (Wang et al., 2018b; Mullenbach et al., 2018; Vu et al., 2020), we report the results of our experiments using macro- and micro-averaged AUC (Area Under the ROC Curve). In a multi-class classification problem, the macro-average computes the metric (AUC in our case) for each class independently and then averages it across classes. This gives the same weight to all classes regardless of possible imbalances in the data. Micro-averaging, on the other hand, computes the average score over all samples, giving the same weight to each sample rather than to each class.

4 Dataset

In this work, we use the widely-used MIMIC-III dataset (Johnson et al., 2016). This dataset contains medical information in various forms, however, as in previous studies (Wang et al., 2018b; Mullenbach et al., 2018; Vu et al., 2020), we consider exclusively the discharge summaries for ICD coding. Discharge summaries are medical notes created by doctors at the end of a stay in a medical facility and contain all the information about the stay. In the MIMIC-III dataset, the length of the discharge summaries after tokenization ranges from 78 to 18,429 tokens with a mean length of 2,740 tokens and a median of 2,500. Each of these discharge summaries has associated to it one or more ICD codes from the ICD-9 taxonomy, with an average of 13.15 ICD codes per summary. Therefore, ICD coding is a multi-label classification task.

The MIMIC-III dataset consists of 52,722 discharge summaries with a total of 8,921 unique ICD codes. However, most of the codes are very infrequent, and therefore, existing work (Wang et al., 2018b; Mullenbach et al., 2018; Vu et al., 2020) narrows down the task to finding only the 50 most frequent ICD codes. We follow this strategy and use the reduced dataset, sometimes referred to as MIMIC-III-50. This dataset consists of a training set of 8,067 samples, a validation set of 1,574 samples and a test set of 1,730 samples. This data split is aligned with previous work, and thus, our results are directly comparable to those in the existing literature.

4.1 Pre-processing

We pre-process the discharge summaries from the MIMIC-III dataset following the method proposed by Mullenbach et al. (2018), which is also used by other recent work (Vu et al., 2020). This way, we convert all the text to lower case and we remove all numbers. However, we do not remove infrequent words as in (Mullenbach et al., 2018) since BERT uses WordPiece for tokenizing and hence, it does not suffer from out-of-vocabulary terms.

5 Model

Discharge summaries are longer than the maximum length accepted by PubMedBERT such that it fits in the memory of a modern GPU and thus, we need to split the summaries into pieces of text. In order to process more than one piece of text per summary we adopt an encoder-decoder structure, where the encoder and the decoder are trained separately. This way, the encoder is the BERT model that maps the different pieces of text to vector representations. These vector representations are then combined and decoded into ICD codes by the decoder, which can be any kind of model.

5.1 Encoder

We use PubMedBERT as the encoder of our model, as described in Section 3. We run our experiments on TITAN RTX GPUs with 24 GB of memory, where we can fit PubMedBERT with a maximum sequence length of 512 tokens.[2] We devise five different strategies to split the text of the discharge summaries:

- *Front*: First 512 tokens of the summary.
- *Back*: Last 512 tokens of the summary.
- *Mixed*: First 256 and the last 256 tokens of the summary.

[1] https://microsoft.github.io/BLURB/leaderboard.html

[2] Note that even if we could fit sequences of 1024 or 2048 tokens, they would still be shorter than the mean and median sequence length of the summaries.

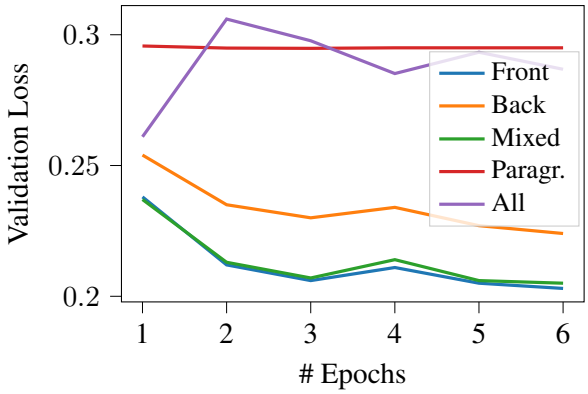

Figure 1: Validation losses for PubMED-BERT trained on different parts of the text.

- *All*: Split the whole discharge summary into consecutive chunks of 512 tokens; since summaries are of different length, each summary is split in a different number of chunks with the last chunk being possibly shorter.
- *Paragraph*: Given that the discharge summaries consist of named paragraphs, we select the 200 most frequent paragraphs, i.e., those that are present most often in the discharge summaries, each with a maximum length of 512 tokens.

PubMedBERT has been pretrained on the masked language modeling task, and therefore, it can produce generic representations of the input text. To fine-tune this model for the ICD coding task without exceeding the memory constraints we can feed only one chunk of text at a time. This way, we fine-tune five different instances of the PubMedBERT model, one per splitting strategy, using a batch size of 1 (to ensure the model fits in memory) and a learning rate of $5e^{-4}$. In each case, the model receives as input a piece of text of a maximum length of 512 tokens and it is trained to predict the ICD codes of the corresponding discharge summary. Note that while the text of *front*, *back* and *mixed* corresponds always to the same part of the discharge summary, when fine-tuning the model on the *paragraph* and *all* splits, each training example consists of only one paragraph or chunk, respectively. Therefore, there is no alignment across training examples (each training example comes from a different section of a discharge summary), which introduces noise to the training.

Figure 1 depicts the validation losses after 6 epochs of training for each of the trained models. For *front*, *back* and *mixed*, we see that the validation loss decreases quickly during the first three epochs and then, it slowly stabilizes. However, for *paragraph* and *all*, the validation loss stays constant, which indicates that the model is failing to learn; in other words, the lack of alignment between training samples makes the task of ICD coding too challenging for the model to learn meaningful representations of the input text.

5.2 Decoder

If we consider only one part of the text at a time, PubMedBERT can directly make a prediction on the ICD codes for the corresponding summary, as done during fine-tuning. However, in order to use the information from different pieces of text, we need a decoder capable of combining the information from several encodings. This way, the decoder receives as input one or several encoded representations (from the same discharge summary) generated by PubMedBERT during the encoding stage and outputs a vector of probabilities for the 50 ICD codes. For the decoder architecture, we consider a linear layer, multi-layer perceptrons (MLPs) and transformers.

In all cases, the decoders are trained with binary cross entropy loss with logits. We use a batch size of 32, a learning rate of $1e^{-4}$ with linear decay for 30 epochs and weight decay with $\lambda = 1e^{-3}$. We train for a maximum of 100 epochs with early stopping on the validation set.

Linear layer Our simplest decoder consists of a linear layer that takes as input a concatenation of the encoding vectors (of size 768 each); when only one chunk is considered, the input is just one encoding vector. The output of this linear layer is the probability vector for the ICD codes.

Multi Layer Perceptron We consider two variants of MLP-architectures, flat and parallel. In the flat architecture, the input is the concatenation of the encodings, as for the linear layer. This vector is passed through two non-linear layers, which produce intermediate representation of size 768 and 512 respectively, and then to a final linear layer that outputs the probabilities of the 50 ICD codes. In the parallel architecture, each of the input encodings is processed by a different dense layer, each of which produces an output of size $768/n$, where n is the number of input encodings. These intermediate representations are concatenated and passed through two additional non-linear layers, with the same sizes as in the flat architecture.

Each of the non-linear layers includes layer normalization (Ba et al., 2016), PReLU activation (He et al., 2015), and dropout (Srivastava et al., 2014) with $p = 0.1$.

Transformer This decoder takes as input the encodings and treats each of them as a token of dimensionality 768. These tokens are passed through a transformer layer with 8 attention heads. The output of this transformer layer is of the same size as the input, i.e., a set of tokens of 768 elements. The tokens are then concatenated and passed through an MLP of the same structure as the *flat* MLP described above.

6 Results

We pose six research questions regarding the different strategies to encode and decode discharge summaries using a BERT-based encoder. In our experiments, we fix the random seed so that all the results are comparable.

How much does fine-tuning the encoder help decoding?

Here, we consider only the PubMedBERT models fine-tuned on *front*, *back* and *mixed* data, since they were the only ones to learn during fine-tuning, as shown in Section 5.1. To investigate the impact of this fine-tuning step on decoding performance, we use a simple linear layer which receives as input the concatenation of the encodings of the *front*, *back* and *mixed* chunks. Each of these pieces of text is encoded by the PubMedBERT model trained on that piece of text, i.e., we use three different encoders. We study the difference in performance for three different training points of the encoders: not fine-tuned, fine-tuned for three epochs and fine-tuned for six epochs. The results are detailed in Table 1.

Epochs	Macro AUC	Micro AUC
None	55.76	69.55
3	81.47	86.00
6	**83.00**	**86.98**

Table 1: Performance for different number of training epochs when combining the *front*, *back* and *mixed* chunks with a linear decoder.

These results show that fine-tuning the encoder significantly improves the decoding performance and that the best performance is obtained after six epochs. In fact, the difference between fine-tuning for six epochs and not fine-tuning is as large as 27.24 points for the Macro AUC score and 17.43 points for the Micro AUC score. We observed the same pattern in all of our experiments, and therefore, in the following we will only present results with the encoder fine-tuned for six epochs, unless stated otherwise.

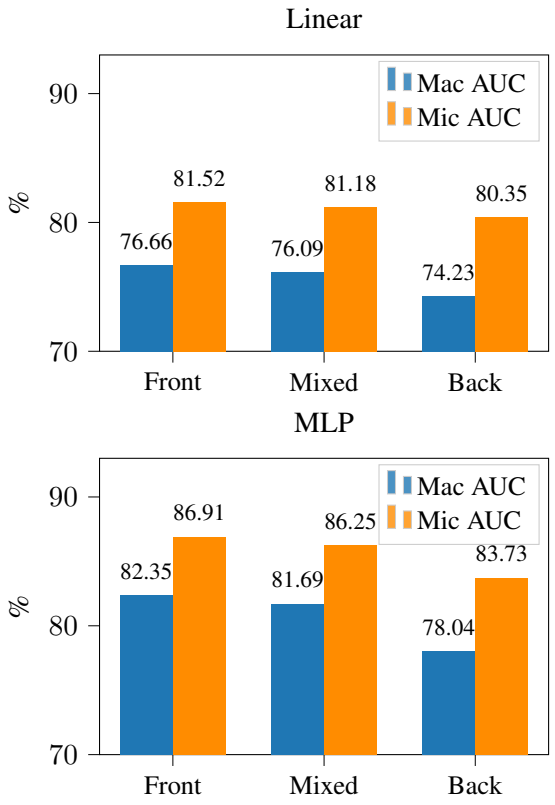

Figure 2: Performance of a linear layer (top) and a non-linear MLP (bottom) on the *front*, *back* and *mixed* encodings.

Which of the three pieces of text, *front*, *back* or *mixed*, contains the most relevant information for ICD coding?

We experiment with a linear and a flat MLP decoder and apply these models to the encodings of each of the three chunks of text separately, i.e., *front*, *back* and *mixed*. We report the results in Figure 2.

We see that *front*, i.e., the first 512 tokens of the discharge summary yields the best performance, both when the decoder is a linear layer and an MLP. Although slightly inferior, the *mixed* chunk produces competitive scores while when using an MLP the AUC scores are more than 3 points lower for *back* than for *front*. Furthermore, using as decoder an MLP improves the performance significantly over using a linear layer; with the *front* non-linear

model performing comparably to the combination of the three chunks with a linear decoder, as reported in the previous section, Table 1.

This naturally raises the question of whether the combination of the chunks yields an improvement. To study this, we use the same non-linear MLP architecture as in Figure 2 (bottom) on 1) the concatenation of the encodings of *front* and *back* and 2) the concatenation of the three encodings, *front*, *back* and *mixed*. We report the results in Table 2.

Model	Mac. AUC	Mic. AUC
Front-Back	83.70	88.11
Front-Back-Mixed	**84.42**	**88.58**

Table 2: Performance of combining the *front*, *back* and *mixed* chunks using a two-layer flat MLP decoder.

These results show that combining *front* and *back* improves performance in comparison to using only *front*. As it may be expected, adding the mixed paragraph, which contains redundant information, produces only a small improvement. Overall, the combination of the three chunks produces an improvement of 2.07 points for Macro AUC and 1.67 points for Micro AUC over using only *front*. Given the larger input, these models have more parameters than the ones using only one of the chunks, which could partly explain the improvement, especially when adding redundant information, i.e., the *mixed* chunk. This result leads us to investigate the influence of the decoder architecture.

How does the architecture of the decoder impact performance?

Here, we consider flat MLP, parallel MLP and transformer decoders on the combination of *front*, *back* and *mixed*. For each of these architectures, we evaluate three different sizes: Base, Large and X-Large, where the difference between these sizes is only the number of layers and the size of the internal representations. This way, our experiments aim at discerning whether the structure of the decoder, the number of parameters, or both, influence the performance of the ICD coding model. Table 3 details the results of these experiments.

None of the models considered obtains a performance significantly higher than the others, with the largest difference across Macro and Micro AUC scores being of only 0.28 and 0.57 points, respectively. This result is surprising since, given the complexity of the task, it could be expected that larger

Model	AUC Mac.	AUC Mic.
Flat ($1.5M$)	84.42	88.58
Flat L ($3M$)	84.30	88.45
Flat XL ($7M$)	84.30	88.47
Parallel ($1M$)	84.45	88.65
Parallel L ($2M$)	84.23	88.48
Parallel XL ($3M$)	84.51	88.49
Transformer ($6.5M$)	84.30	88.49
Transformer L ($14M$)	84.27	88.45
Transformer XL ($18M$)	84.29	88.08

Table 3: Performance of different decoder architectures for the combination of *front*, *back* and *mixed*, the number of parameters of each model is specified in parenthesis.

and more sophisticated decoders would perform better. Notwithstanding, the saturation in performance suggests that all the information available in the input of the decoder is successfully extracted by every model, regardless of its complexity. This in turn indicates that the performance of the whole encoder-decoder model is limited by the reduced amount of text that is given as input (only the beginning and the end of the discharge summaries). Therefore, we next consider providing larger portions of text from the discharge summaries as input.

Is dividing the discharge summaries by paragraphs a good splitting strategy?

By splitting the discharge summaries into paragraphs we take into account information from a larger body of text than by using the front and the back. The main disadvantage of this approach is that the encoder fails to converge during fine-tuning. Here, we test the hypothesis of whether the decoder can compensate the lack of fine-tuning of the encoder and, by leveraging the larger amount of information available, reach competitive performance. We encode the 200 most frequent paragraphs using the PubMedBERT model fine-tuned on *paragraph* data, although due to lack of convergence during fine-tuning, we observed very similar results when using the not fine-tuned version.

Since not all the discharge summaries contain the same paragraphs, there is a misalignment between samples. For this reason, here we consider only the transformer decoder; the self-attention modules of the transformer should be able to cope with the misalignment better than the other architectures. We consider the transformer decoders (Base, Large and X-Large) from the previous section. Now, the

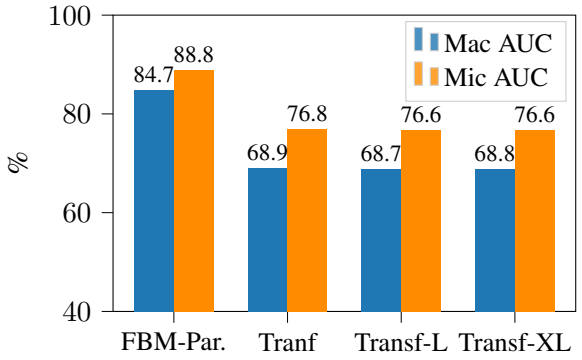

Figure 3: Comparison of front-back-mixed parallel (FBM-Par.) and three sizes of transformer decoders (Transf) on *paragraph* data.

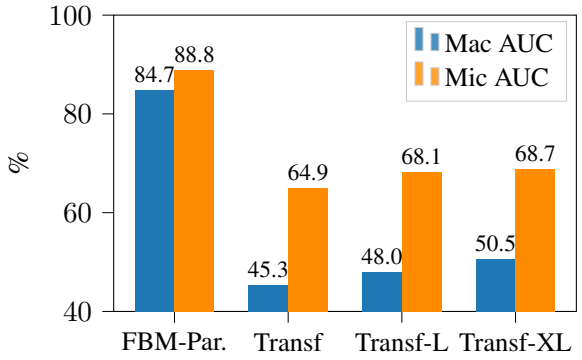

Figure 4: Comparison of front-back-mixed parallel (FBM-Par.) and three sizes of transformer decoders (Transf) on *all* data.

transformer decoder receives 200 encoded representations, one per paragraph. Given this large number of input representations or tokens, we aggregate the output of the transformers by taking the mean over the representations produced for all the paragraphs[3].

In Figure 3, we compare these *paragraph* decoders to the Parallel MLP model on the *front*, *back* and *mixed* chunks from the previous section.

We see that dividing the discharge summaries into paragraphs greatly under-performs in comparison to using the beginning and end of the summaries encoded by fine-tuned PubMedBERT models. This result partly rejects the hypothesis that the decoder can benefit from a larger unstructured input. Next, we continue investigating this hypothesis by feeding the decoder with the complete discharge summaries following the *all* strategy.

How does splitting the complete summaries in consecutive chunks perform?

We split the whole text of each discharge summary into consecutive chunks of 512 tokens (the last chunk of each summary may be smaller). We encode these chunks using the PubMedBERT model fine-tuned on *all* data; as before, we observed very similar results with the not fine-tuned model. The encodings are then fed into the decoder. Again, the varying size of the discharge summaries produces misalignment across examples. Therefore, we consider only the transformer decoders (Base, Large and X-Large). We report the results of this experiment in Figure 4.

The largest transformer model (XL) performs the best of the three models on *all* data. Nevertheless, its 50.5% Macro and 68.7% Micro AUC scores are much lower than the results obtained by the *front-back-mixed*. In fact, splitting the text into chunks of the same size performs the worst among all the methods that we have investigated. These results confirm that the decoder cannot compensate the lack of convergence during the fine-tuning of the encoder and points at the encoder as the main responsible for the model's performance.

How do our results compare to the state-of-the-art?

Finally, in Table 4 we compare one of our best performing BERT-ICD models, the *front-back-mixed* Parallel model, with the existing state-of-the art models for ICD coding on the MIMIC-III dataset. In particular, we compare against the condensed memory networks (C-MemNN) by Prakash et al. (2017), LEAM by Wang et al. (2018b), CAML and DR-CAML by Mullenbach et al. (2018), MSATT-KG by Xie et al. (2019) and the Label Attention model by Vu et al. (2020). We report the performance of each model as provided in the corresponding original work.

Model	AUC Mac.	AUC Mic.
C-MemNN	83.3	-
LEAM	88.1	91.2
CAML	87.5	90.9
DR-CAML	88.0	90.2
MSATT-KG	91.4	93.6
Label Attention	**92.1**	**94.6**
BERT-ICD	84.45	88.65

Table 4: Comparison of different state-of-the-art models for ICD coding.

[3]We experimented with other aggregation techniques like max pooling and large MLPs obtaining very similar results.

We see that although our BERT-based ICD coding model is competitive with some of the state-of-the-art models, there is still a substantial gap between the best performing model from Vu et al. (2020), and our BERT-ICD model.

7 Discussion

Automatic ICD coding from discharge summaries using transformer models has proven to be challenging. Discharge summaries are very long documents and thus, they need to be divided into chunks in order to be entirely processed by BERT-like models.This way, a decoder is necessary to combine the representations of each chunk, which are independently generated by the BERT encoder. We have shown that for these representations to be meaningful the encoder needs to be fine-tuned on the ICD coding task. It is straight-forward to fine-tune a BERT encoder such as PubMedBERT using specific parts of the summary, e.g., the beginning or the end. However, in our experiments, fine-tuning PubMedBERT on excerpts extracted from different parts of the text, i.e., *paragraph* and *all*, prevented convergence due to the lack of alignment between samples, i.e., due to each training sample containing information from a different section of a discharge summary. Furthermore, our results show that the decoder, regardless of its architecture, cannot compensate for lack of convergence during the fine-tuning of the encoder.

On the other hand, our best BERT-ICD model reaches competitive performance using only $1,024$ tokens (*front* and *back*), which represents a significantly smaller portion of text than state-of-the-art models, based on CNNs and RNNs. Unlike BERT, CNN and RNN models can extract information from texts of any length without needing to split them, which allows for end-to-end training over long pieces of text. Mullenbach et al. (2018) found that the performance of their convolutional attention model benefits from longer input texts until a length of between $2,500$ and $6,500$ words, and Vu et al. (2020) use up to $4,000$ words as input. Our model combines encodings from the beginning and the end of the discharge summary, and reaches better performance in that case than when it processes either of those portions of text alone. This supports the statement that including more text improves ICD coding. All of these results suggest that the difficulty of fine-tuning a BERT encoder on long pieces of text is the main bottleneck for performance and the reason for the existing gap with state-of-the-art models for ICD coding.

One of the main advantages of transformer models over CNNs and RNNs is that they can handle long-range dependencies. Hence, if longer text could be fed at once into a BERT encoder it would be possible to find relationships and patterns over longer spans of text. It is therefore likely that advances either in terms of hardware, i.e., larger GPU memories allowing for longer pieces of text to be processed at once; or in compressing BERT-like models, e.g., distillation, will progressively close the gap with the state-of-the-art, following the same trend of other areas of NLP. On top of that, we consider that the two most promising directions for future research on BERT-based ICD coding are: 1) devising strategies to fine-tune the encoder over longer spans of text, e.g., building an ensemble of models where each of them is trained on one section of the text; 2) improving the methods to aggregate encodings from different parts of the document.

Finally, to deploy automatic ICD coding systems in the real world, it is important that their decisions can be explained. Explaining transformer models is currently a field of active research, and although there exist important concerns about the interpretability of attention distributions in transformers (Brunner et al., 2019; Pruthi et al., 2020), methods based on gradient attribution (Pascual et al., 2020) or on attention flow (Abnar and Zuidema, 2020) can provide insights on their decision-making. A BERT-based ICD coding system could directly benefit from this field of research and eventually provide explanations together with its ICD code predictions.

8 Conclusion

Contrary to what is common in most NLP tasks, the transformer architecture is not the state-of-the-art in assigning ICD codes to discharge summaries. In this work, we have presented a thorough study of the performance of BERT-based models on this task and we have identified the length of the discharge summaries as the main obstacle holding back their performance. Our work sets a solid foundation for further research on ICD coding and suggests that overcoming the exposed limitations of BERT-based models is likely to lead to a new state-of-the-art. Furthermore, we believe that the interpretability of ICD coding models is an interesting avenue for

future work, which can benefit from a large body of existing research.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197.

Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*.

Anand Avati, Kenneth Jung, Stephanie Harman, Lance Downing, Andrew Ng, and Nigam H Shah. 2018. Improving palliative care with deep learning. *BMC medical informatics and decision making*, 18(4):55–64.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Tal Baumel, Jumana Nassour-Kassis, Raphael Cohen, Michael Elhadad, and Noémie Elhadad. 2018. Multi-label classification of patient notes: Case study on icd code assignment. In *AAAI Workshops*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. In *International Conference on Learning Representations*.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic icd coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3105–3114.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference*, pages 301–318. PMLR.

Luciano RS de Lima, Alberto HF Laender, and Berthier A Ribeiro-Neto. 1998. A hierarchical approach to the automatic categorization of medical documents. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 132–139.

Joshua C Denny, Marylyn D Ritchie, Melissa A Basford, Jill M Pulley, Lisa Bastarache, Kristin Brown-Gentry, Deede Wang, Dan R Masys, Dan M Roden, and Dana C Crawford. 2010. Phewas: demonstrating the feasibility of a phenome-wide scan to discover gene–disease associations. *Bioinformatics*, 26(9):1205–1210.

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.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *arXiv preprint arXiv:2007.15779*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Sarvnaz Karimi, Xiang Dai, Hamed Hassanzadeh, and Anthony Nguyen. 2017. Automatic diagnosis coding of radiology reports: a comparison of deep learning and conventional classification methods. In *BioNLP 2017*, pages 328–332.

Leah S Larkey and W Bruce Croft. 1996. Combining classifiers in text categorization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 289–297.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Fei Li and Hong Yu. 2020. Icd coding from clinical text using multi-filter residual convolutional neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *NAACL-HLT*.

Damian Pascual, Gino Brunner, and Roger Wattenhofer. 2020. Telling bert's full story: from local attention to global aggregation. *arXiv preprint arXiv:2004.05916*.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, and Noémie Elhadad. 2014. Diagnosis code assignment: models and evaluation metrics. *Journal of the American Medical Informatics Association*, 21(2):231–237.

Aaditya Prakash, Siyuan Zhao, Sadid Hasan, Vivek Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2017. Condensed memory networks for clinical diagnostic inferencing. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. 2011. Multiparameter intelligent monitoring in intensive care ii (mimic-ii): a public-access intensive care unit database. *Critical care medicine*, 39(5):952.

Elyne Scheurwegs, Boris Cule, Kim Luyckx, Léon Luyten, and Walter Daelemans. 2017. Selecting relevant features from the electronic health record for clinical code prediction. *Journal of biomedical informatics*, 74:92–103.

Haoran Shi, Pengtao Xie, Zhiting Hu, Ming Zhang, and Eric P Xing. 2017. Towards automated icd coding using deep learning. *arXiv preprint arXiv:1711.04075*.

Congzheng Song, Shanghang Zhang, Najmeh Sadoughi, Pengtao Xie, and Eric P Xing. 2020. Generalized zero-shot text classification for icd coding. In *IJCAI*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for icd coding from clinical text. *arXiv preprint arXiv:2007.06351*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018a. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018b. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. 2019. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 649–658.