BERT FOR SEQUENCE-TO-SEQUENCE MULTI-LABEL TEXT CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the BERT language representation model and the sequence generation model with BERT encoder for multi-label text classification task. We experiment with both models and explore their special qualities for this setting. We also introduce and examine experimentally a mixed model, which is an ensemble of multi-label BERT and sequence generating BERT models. Our experiments demonstrated that BERT-based models and the mixed model, in particular, outperform current baselines in several metrics achieving state-of-the-art results on three well-studied multi-label classification datasets with English texts and two private Yandex Taxi datasets with Russian texts.

1 INTRODUCTION

Multi-label text classification (MLTC) is an important natural language processing task with many applications, such as document categorization, automatic text annotation, protein function prediction (Wehrmann et al., 2018), intent detection in dialogue systems, and tickets tagging in client support systems (Molino et al., 2018). In this task, text samples are assigned to multiple labels from a finite label set.

In recent years, it became clear that deep learning approaches can go a long way toward solving text classification tasks. However, most of the widely used approaches in MLTC tend to neglect correlation between labels. One of the promising yet fairly less studied methods to tackle this problem is using sequence-to-sequence modeling. In this approach, a model treats an input text as a sequence of tokens and predict labels in a sequential way taking into account previously predicted labels. Nam et al. (2017) used Seq2Seq architecture with GRU encoder and attention-based GRU decoder, achieving an improvement over a standard GRU model (Cho et al., 2014) on several datasets and metrics. Yang et al. (2018b) continued this idea by introducing Sequence Generation Model (SGM) consisting of BiLSTM-based encoder and LSTM decoder coupled with additive attention mechanism (Bahdanau et al., 2014).

In this paper, we argue that the encoder part of SGM can be successfully replaced with a heavy language representation model such as BERT (Devlin et al., 2018). We propose Sequence Generating BERT model (BERT+SGM) and a mixed model which is an ensemble of vanilla BERT and BERT+SGM models. We show that BERT+SGM model achieves decent results after less than a half of an epoch of training, while the standard BERT model needs to be trained for 5-6 epochs just to achieve the same accuracy and several dozens epochs more to converge. On public datasets, we obtain 0.4%, 0.8%, and 1.6% average improvement in miF_1, maF_1, and accuracy respectively in comparison with BERT. On datasets with hierarchically structured classes, we achieve 2.8% and 1.5% average improvement in maF_1 and accuracy.

Our main contributions are as follows:

1. We present the results of BERT as an encoder in the sequence-to-sequence framework for MLTC datasets with and without a given hierarchical tree structure over classes.
2. We introduce and examine experimentally a novel mixed model for MLTC.
3. We fine-tune the vanilla BERT model to perform multi-label text classification. To the best of our knowledge, this is the first work to experiment with BERT and explore its particular properties for the multi-label setting and hierarchical text classification.

4. We demonstrate state-of-the-art results on three well-studied MLTC datasets with English texts and two private Yandex Taxi datasets with Russian texts.

2 Related Work and Preliminaries

Let us consider a set $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}$ consisting of N samples that are assumed to be identically and independently distributed following an unknown distribution $P(\mathbf{X}, \mathbf{Y})$. *Multi-class* classification task aims to learn a function that maps inputs to the elements of a label set $\mathcal{L} = \{1, 2, \ldots, L\}$, i.e. $\mathcal{Y} = \mathcal{L}$. In *multi-label* classification, the aim is to learn a function that maps inputs to the subsets of $\mathcal{L}$, i.e. $\mathcal{Y} = 2^{\mathcal{L}}$. In text classification tasks, $\mathcal{X}$ is a space of natural language texts.

A standard pipeline in deep learning is to use a base model that converts a raw text to its fixed-size vector representation and then pass it to a classification algorithm. Typical architectures for base models include different types of recurrent neural networks (Hochreiter & Schmidhuber, 1997; Cho et al., 2014), convolutional neural networks (Kim, 2014), hierarchical attention networks (Yang et al., 2016), and other more sophisticated approaches. These models consider each instance $\boldsymbol{x}$ as a sequence of tokens $\boldsymbol{x} = [w_1, w_2, \ldots, w_T]$. Each token w_i is then mapped to a vector representation $\boldsymbol{u}_i \in \mathbb{R}^H$ thus forming an embedding matrix $U^{T \times H}$ which can be initialized with pre-trained word embeddings (Mikolov et al., 2013; Pennington et al., 2014). Moreover, recent works show that it is possible to pre-train entire language representation models on large corpora of texts in a self-supervised way. Newly introduced models providing context-dependent text embeddings, such as ELMo (Peters et al., 2018), ULMFiT (Howard & Ruder, 2018), OpenAI GPT (Radford et al., 2018), and BERT (Devlin et al., 2018) significantly improved previous state-of-the-art results on various NLP tasks. Among the most recent works, XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019) models improve these results further after overcoming some limitations of original BERT.

A novel approach to take account of dependencies between labels is using Seq2Seq modeling. In this framework that first appeared in the neural machine translation field (Sutskever et al., 2014), we generally have source input $\mathbf{X}$ and target output $\mathbf{Y}$ in the form of sequences. We also assume there is a hidden dependence between $\mathbf{X}$ and $\mathbf{Y}$, which can be captured by probabilistic model $P(\mathbf{Y}|\mathbf{X}, \theta)$. Therefore, the problem consists of three parts: modeling the distribution $P(\mathbf{Y}|\mathbf{X}, \theta)$, learning the parameters θ, and performing the inference stage where we need to find $\hat{\mathbf{Y}} = \arg_{\mathbf{Y}} \max P(\mathbf{Y}|\mathbf{X}, \theta)$.

Nam et al. (2017) have shown that after introducing a total order relation on the set of classes $\mathcal{L}$, the MLTC problem can be treated as sequence-to-sequence task with $\mathbf{Y}$ being the ordered set of relevant labels $\{l_1, l_2, \ldots, l_M\} \subseteq \mathcal{L}$ of an instance $\mathbf{X} = [w_1, w_2, \ldots, w_T]$. The primary approach to model sequences is decomposing the joint probability $P(\mathbf{Y}|\mathbf{X}, \theta)$ into M separate conditional probabilities. Traditionally, the left-to-right (L2R) order decomposition is:

$$P(l_1, l_2, \ldots, l_M|\boldsymbol{x}) = \prod_{i=1}^{M} P(l_i|l_{1:i-1}, \boldsymbol{x}) \tag{1}$$

Wang et al. (2016) demonstrated that the label ordering in (1) effects on the model accuracy, and the order with descending label frequencies results in a decent performance on image datasets. Alternatively, if an additional prior knowledge about the relationship between classes is provided in the form of a tree hierarchy, the labels can also be sorted in topological order with a depth-first search performed on the hierarchical tree. Nam et al. (2017) argued that both orderings work similarly well on text classification datasets.

A given hierarchical structure over labels forms a particular case of text classification task known as hierarchical text classification (HTC). Such an underlying structure over the set of labels can help to discover similar classes and transfer knowledge between them improving the accuracy of the model for the labels with only a few training examples (Srivastava & Salakhutdinov, 2013). Most of the researchers' efforts to study HTC were dedicated to computer vision applications (Wang et al., 2016; Yan et al., 2015; Srivastava & Salakhutdinov, 2013; Salakhutdinov et al., 2011), but many of these studies potentially can be or have already been adapted to the field of natural language texts. Among the most recent works, Peng et al. (2018) proposed a Graph-based CNN architecture with a hierarchical regularizer, and Wehrmann et al. (2018) argued that mixing an output from a *global* classifier and the outputs from all layers of a *local* classifier can be beneficial to learn hierarchical

dependencies. It was also shown that reinforcement learning models with special award functions can be applied to learn non-trivial losses (Yang et al., 2018a; Mao et al.).

3 BERT-BASED MODELS FOR MULTI-LABEL TEXT CLASSIFICATION

3.1 BERT MODEL AS A TEXT ENCODER

BERT (Bidirectional Encoder Representations from Transformers) is a recently proposed language representation model for obtaining text embeddings. BERT was pre-trained on unlabelled texts for masked word prediction and next sentence prediction tasks, providing deep bidirectional representations. For classification tasks, a special token `[CLS]` is put to the beginning of the text and the output vector of the token `[CLS]` is designed to correspond to the final text embedding. The pre-trained BERT model has proven to be very useful for transfer learning in multi-class and pairwise text classification. Fine-tuning the model followed by one additional feedforward layer and softmax activation function was shown to be enough for providing state-of-the-art results on a downstream task (Devlin et al., 2018).

For examining BERT on the multi-label setting, we change activation function after the last layer to sigmoid so that for each label we predict their probabilities independently. The loss to be optimized will be adjusted accordingly from cross-entropy loss to binary cross-entropy loss.

3.2 BERT ENCODER FOR SEQUENCE GENERATION

In sequence generation model (Yang et al., 2018b), the authors use BiLSTM as an encoder with pre-trained word embeddings of dimension $d = 512$. For a raw text $\boldsymbol{x} = [w_1, w_2, \ldots, w_T]$ each word w_i is mapped to its embedding $\boldsymbol{u}_i \in \mathbb{R}^d$, and contextual word representations are computed as follows:

$$\overrightarrow{\boldsymbol{h}}_i = \overrightarrow{\text{LSTM}}(\overrightarrow{\boldsymbol{h}}_{i-1}, \boldsymbol{u}_i), \quad \overleftarrow{\boldsymbol{h}}_i = \overleftarrow{\text{LSTM}}(\overleftarrow{\boldsymbol{h}}_{i+1}, \boldsymbol{u}_i), \quad \boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i] \tag{2}$$

After that, the decoder's zeroth hidden state is initialized as $\boldsymbol{s}_0 = f_{init}([\overrightarrow{\boldsymbol{h}}_0; \overleftarrow{\boldsymbol{h}}_T])$.

We propose to use the outputs of the last transformer block in BERT model as vector representations of words and the embedding of the token `[CLS]` produced by BERT as the initial hidden state of the decoder. We also use a simple dot-product attention mechanism which in our setting showed similar performance as additive attention, but resulted in less number of parameters to learn. The process we follow to calculate decoder's hidden states $\boldsymbol{\alpha}_t$ and the attention scores $\boldsymbol{\alpha}_t$ is described in Algorithm 1 and illustrated in Figure 1. The weight matrices $\boldsymbol{W}_{init}, \boldsymbol{W}_o, \boldsymbol{W}_d, \boldsymbol{V}_d$ are all learnable parameters. It is also worth mentioning that we do not freeze BERT parameters so that they can also be fine-tuned in the training process.

In order to maximize the total likelihood of the produced sequence, we train the final model to minimize the cross-entropy objective loss for a given $\boldsymbol{x}$ and ground-truth labels $\{l_1^*, l_2^*, \ldots, l_k^*\} \in \mathcal{L}$:

$$\mathcal{L}_{\text{CE}}(\theta) = -\sum_{i=1}^{k} \log P(l_i^* | \boldsymbol{x}, l_{1:i-1}^*, \theta) \tag{3}$$

In the inference stage, we can compute the objective 3 replacing ground-truth labels with predicted labels. To produce the final sequence of labels, we perform a beam search following the work (Wiseman & Rush, 2016) to find candidate sequences that have the minimal objective scores among the paths ending with the `<EOS>` token.

3.3 MIXED MODEL

In further experiments, we mainly test standard BERT and sequence generating BERT models. From our experimental results that will be demonstrated later on, we concluded that BERT and BERT+SGM may each have their advantages and drawbacks on different datasets. Therefore, to make the models alleviate each other's weaknesses, it might be reasonable to combine them. Our error analysis on a number of examples has shown that in some cases, BERT can predict excess

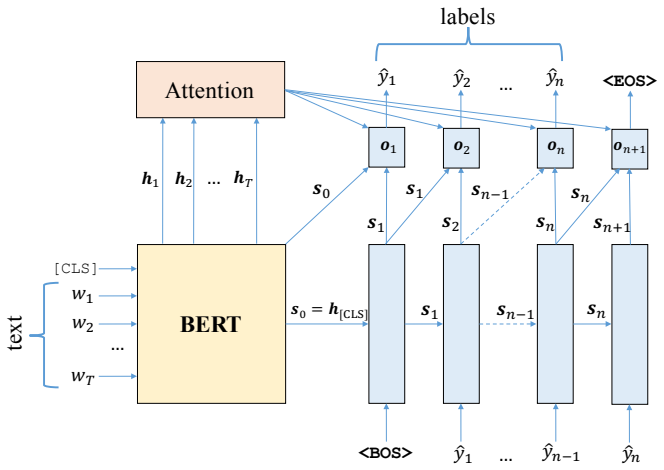

Figure 1: BERT + SGM. An overview of the model.

Algorithm 1 BERT + SGM

$\mathcal{L}_{pred} \leftarrow \{\}$
$\mathcal{L} \leftarrow \{1, 2, \ldots, L, \texttt{<EOS>}\}$
$[\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_T], \boldsymbol{h}_{\texttt{[CLS]}} \leftarrow \text{BERT}(\boldsymbol{x})$
$s_0 \leftarrow \boldsymbol{W}_{init} \boldsymbol{h}_{\texttt{[CLS]}}$
$\hat{y}_0 \leftarrow \texttt{<BOS>}$
$t \leftarrow 0$
while $\hat{y}_t \neq \texttt{<EOS>}$ **do**
  $t \leftarrow t + 1$
  $\boldsymbol{\alpha}_t = \text{softmax}([\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_T]^T \boldsymbol{s}_{t-1})$
  $\boldsymbol{c}_t \leftarrow \sum_{i=1}^{T} \alpha_{ti} \boldsymbol{h}_i$
  $\boldsymbol{s}_t \leftarrow \text{LSTM}(\boldsymbol{s}_{t-1}, [\hat{y}_{t-1}; \boldsymbol{c}_{t-1}])$
  $\boldsymbol{o}_t \leftarrow \boldsymbol{W}_o \tanh(\boldsymbol{W}_d \boldsymbol{s}_t + \boldsymbol{V}_d \boldsymbol{c}_t)$
  for $i \in \{0, 1, 2, \ldots, L\}$ **do**
    if $\mathcal{L}_i \in \mathcal{L}_{pred}$ **then**
      $I_{ti} \leftarrow -\infty$
    else
      $I_{ti} \leftarrow 0$
  $\boldsymbol{y}_t \leftarrow \text{softmax}(\boldsymbol{o}_t + \boldsymbol{I}_t)$
  $\hat{y}_t \leftarrow \arg\max \boldsymbol{y}_t$
  $\mathcal{L}_{pred} \leftarrow \mathcal{L}_{pred} \cup \{\hat{y}_t\}$

labels while BERT+SGM tends to be more restrained, which suggests that the two approaches can potentially complement each other well.

Another argument in favor of using a hybrid method is that in contrast to the multi-label BERT model, BERT+SGM exploits the information about the underlying structure of labels. Wehrmann et al. (2018) in their work propose HMCN model in which they suggest to jointly optimize both local (hierarchical) and global classifiers and combine their final probability predictions as a weighted average.

Inspired by this idea, we propose to use a mixed model which is an ensemble of multi-label BERT and sequence generating BERT models. A main challenge in creating a mixed model is that the outputs of the two models are quite different. Typically, we do not have access to a probability distribution over the labels in classic Seq2Seq framework. We suggest to tackle this problem by computing the probability distributions produced by the decoder at each stage and then perform element-wise max-pooling operation on them following the idea of the recent paper (Salvador et al., 2018). We should emphasize that using these probabilities to produce final label sets will not necessarily result in the same predictions as the original BERT + SGM model. However, in our experiments, we found that the probability distributions obtained in that way are quite meaningful and with proper prob-

DATASET	N	L	W	C	STRUCTURE	LANGUAGE
RCV1-v2	804 410	103	223.2 ± 206.6	3.2 ± 1.4	Tree	Eng
Reuters-21578	10 787	90	142.2 ± 142.5	1.2 ± 0.7	-	Eng
AAPD	55 840	54	155.9 ± 67.6	2.4 ± 0.7	-	Eng
Y.Taxi Drivers	163 633	374	18.9 ± 22.6	2.1 ± 1.0	Tree	Rus
Y.Taxi Riders	174 590	426	16.2 ± 18.6	3.4 ± 0.8	Tree	Rus

Table 1: Summary of the datasets. N is the number of documents, L is the number of labels, W denotes the average number of words per sample ± SD, and C denotes the average number of labels per sample ± SD.

ability threshold (around 0.4-0.45 for the considered datasets) can yield predictions with accuracy comparable to the accuracy of BERT+SGM model's predictions from the inference stage.

After obtaining probability distributions of both models, we can compute their weighed average to create the final probability distribution vector, as follows:

$$\boldsymbol{p}_{\text{mixed}} = \alpha \boldsymbol{p}_{\text{BERT+SGM}} + (1 - \alpha)\boldsymbol{p}_{\text{BERT}} \tag{4}$$

This probability vector is then used to make final predictions of labels with 0.5 probability threshold. The value of $\alpha \in [0, 1]$ is a trade-off parameter that is optimized on validation set. The final procedure is presented in Algorithm 2.

Algorithm 2 Mixed Model

$\boldsymbol{p}_{\text{BERT}} \leftarrow \text{BERT}(\boldsymbol{x})$
$[\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n] \leftarrow \text{BERT+SGM}(\boldsymbol{x})$
for $l \in \{1, 2, \ldots, L\}$ **do**
  $p^{(l)}_{\text{BERT+SGM}} \leftarrow \max\{y_{1l}, y_{2l}, \ldots, y_{nl}\}$
$\boldsymbol{p}_{\text{mixed}} \leftarrow \alpha \boldsymbol{p}_{\text{BERT+SGM}} + (1 - \alpha)\boldsymbol{p}_{\text{BERT}}$
$\mathcal{L}_{pred} \leftarrow \{l \mid p^{(l)}_{\text{mixed}} \geqslant 0.5\}$

4 EXPERIMENTS

4.1 DATASETS AND PREPROCESSING

We train and evaluate all the models on three public datasets with English texts and two private datasets with Russian texts. The summary of the datasets' statistics is provided in the Table 1. Preprocessing of the datasets included lower casing the texts and removing punctuation. For the baseline TextCNN and SGM models, we used the same preprocessing techniques as in (Yang et al., 2018b).

Reuters Corpus Volume I (RCV1-v2) (Lewis et al., 2004) is a collection of manually categorized 804 410 news stories (after dropping four empty samples from the testing set). There are 103 categories organized in a tree hierarchy, and each text sample is assigned to labels from one or multiple paths in the tree. Since there was practically no difference between topological sorting order and order by frequency (Nam et al., 2017) in multi-path case, we chose to sort the labels from the most common ones to the rarest ones. The training/testing split for this dataset is originally 23,149 in the training set and 781,261 in the testing set (Lewis et al., 2004). While this training/testing split is still used in modern research works (Nam et al., 2013; Mao et al.), in some other works authors have (implicitly) shifted towards using reverse training/testing split (Nam et al., 2017), and several other recent research works (Lin et al., 2018; Yang et al., 2018a;b) started using 802,414 samples for the training set and 1,000 samples for the validation and testing sets. This change of the split might be reasonable due to the inadequate original proportion of the sets in modern realities, yet it makes it difficult to perform an apple-to-apple comparison of different models without their reimplementation. To avoid confusion, we decided to be consistent with the original training/testing split. We also used 10% of the training data for validation.

Reuters-21578 is one of the most commonly used MLTC benchmark datasets with 10,787 articles from Reuters newswire collected in 1987 and tagged with 90 labels. We use the standard ApteMod split of the dataset following the work (Cohen & Singer, 1996).

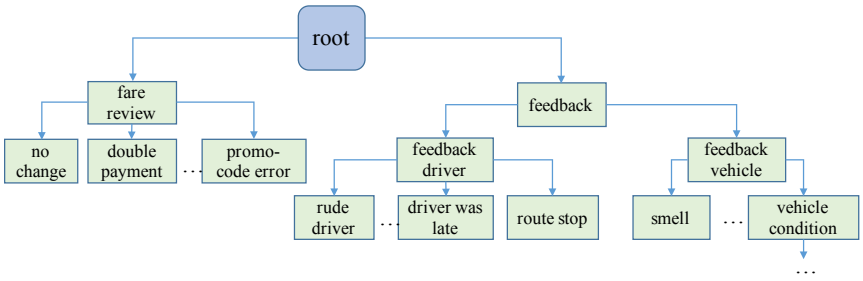

Figure 2: An example of a subtree of the tree hierarchy over classes in *Y.Taxi Riders* dataset.

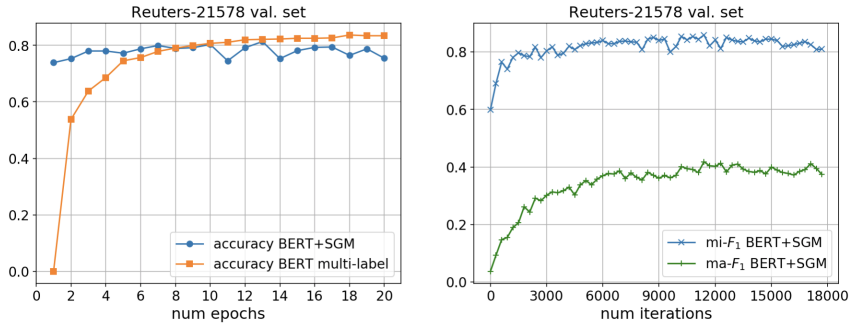

Figure 3: Performance of BERT and BERT+SGM on *Reuters-21578* validation set during training.

Arxiv Academic Paper Dataset (AAPD) is a recently collected dataset (Yang et al., 2018b) consisting of abstracts of 55,840 research papers from `arXiv.org`. Each paper belongs to one or several academic subjects, and the task is to predict those subjects for a paper based on its abstract. The number of categories is 54. We refer the reader to Appendix B for visualization of multi-label BERT embeddings for some of the labels from this dataset.

Riders Tickets from Yandex Taxi Client Support (Y.Taxi Riders) is a private dataset obtained in Yandex Taxi client support system consisting of 174,590 tickets from riders. Initially, the dataset was labeled by Yandex Taxi reviewers with one tag per each ticket sample with an estimated accuracy of labeling around 75-78%. However, using additional information about a tree hierarchical structure over labels, we substituted each label with the corresponding label set with all the parent classes lying in the path between the root node and the label node. After this procedure, we ended up with 426 labels. Since in this task there is only one path in the tree to be predicted, we will explore a natural topological label ordering for this dataset. An example of a subtree of the tree hierarchy is provided in Figure 2.

Drivers Tickets from Yandex Taxi Client Support (Y.Taxi Drivers) is also a private dataset obtained in Yandex Taxi drivers support system which has similar properties with the *Y.Taxi Riders* dataset. In the drivers' version, there are 163,633 tickets labeled with 374 tags.

4.2 Experiment Settings and Baselines

We implemented all the experiments in PyTorch 1.0 and ran the computations on a GeForce GTX 1080Ti GPU. Our implementation is relied on pytorch-transformers library [1].

In the experiments, we used the base-uncased versions of BERT for English texts and the base-cased-multilingual version for Russian texts. Models of both versions output 768-dimensional hidden representation vector. We set batch size to 16. For optimization, we used Adam optimizer (Kingma & Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and learning rate $2 \cdot 10^{-5}$. For the multi-label BERT, we also used the same scheduling of the learning rate as in the original work by Devlin et al. (2018).

[1] `https://github.com/huggingface/pytorch-transformers` (the former name: pytorch-pretrained-bert)

	RCV1-v2				*Reuters-21578*				*AAPD*			
	HA	miF_1	maF_1	ACC	HA	miF_1	maF_1	ACC	HA	miF_1	maF_1	ACC
TextCNN	0.990	0.829	0.456	0.600	0.991	0.851	0.437	0.827	0.974	0.674	0.445	0.364
HMCN	-	0.808	0.546	-	-	-	-	-	-	-	-	-
HiLAP	-	0.833	0.611	-	-	-	-	-	-	-	-	-
EncDec orig.	-	-	-	-	0.996	0.858	0.457	0.828	-	-	-	-
SGM repr.	0.990	0.815	0.428	0.605	0.996	0.788	0.452	0.812	0.974	0.698	0.468	0.372
BERT	**0.992**	0.864	0.556	0.624	**0.997**	**0.899**	**0.534**	**0.857**	0.976	0.713	**0.559**	0.381
BERT+SGM	0.990	0.846	**0.629**	0.602	0.996	0.854	0.467	0.817	0.976	**0.718**	0.496	0.377
Mixed	**0.992**	**0.868**	0.611	**0.631**	**0.996**	**0.900**	**0.533**	**0.858**	**0.977**	**0.719**	0.553	**0.397**

	Y.Taxi Drivers				*Y.Taxi Riders*			
	HA	miF_1	maF_1	ACC	HA	miF_1	maF_1	ACC
TextCNN	0.996	0.610	0.173	0.571	0.994	0.521	0.130	0.381
SGM repr.	0.996	0.629	0.148	0.584	0.993	0.545	0.112	0.399
BERT	0.997	**0.692**	0.226	0.578	0.995	**0.658**	0.153	0.452
BERT+SGM	0.997	0.644	0.196	0.596	**0.997**	0.644	**0.176**	0.465
Mixed	**0.998**	0.681	**0.235**	**0.599**	**0.997**	**0.657**	**0.174**	**0.469**

Table 2: Results on the five considered datasets. Metrics are marked in bold if they contain the highest metrics for the dataset in their ±SD interval.

Following previous research works (Nam et al., 2017), we used hamming accuracy, set accuracy, micro-averaged f_1, and macro-averaged f_1 to evaluate the performance of the models. To be specific, the former two metrics can be computed as $\text{ACC}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \mathbb{1}(\boldsymbol{y} = \hat{\boldsymbol{y}})$ and $\text{HA}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{L}\sum_{j=1}^{L} \mathbb{1}(y_j = \hat{y}_j)$ and are designed to determine the accuracy of the predicted sets as whole. The latter ones are label-based metrics and can be calculated as follows:

$$\text{mi}F_1 = \frac{\sum_{j=1}^{L} 2tp_j}{\sum_{j=1}^{L}(2tp_j + fp_j + fn_j)}, \quad \text{ma}F_1 = \frac{1}{L}\sum_{j=1}^{L}\frac{2tp_j}{2tp_j + fp_j + fn_j} \tag{5}$$

where tp_j, fn_j, and fp_j denote the number of true positive, false positive and false negative predictions for the label j, respectively.

We use a classic convolutional neural network **TextCNN** (Kim, 2014) as a baseline for our experiments. We implemented a two-layer CNN with each layer followed by max pooling and two feed-forward fully-connected layers followed by dropout and batch normalization at the end. Our second baseline model is Sequence Generation Model **SGM** (Yang et al., 2018b), for which we reused the implementation of the authors [2]. For the sake of comparison, we also provide the results of **HMCN** (Wehrmann et al., 2018) and **HiLAP** (Mao et al.) models for hierarchical text classification on *RCV1-v2* dataset adopted from the work (Mao et al.). For *Reuters-21578* dataset, we also included the results of the **EncDec** model (Nam et al., 2017) from the original paper on sequence-to-sequence approach to MLTC.

4.3 Results and Discussion

We present the results of the suggested models and baselines on the five considered datasets in Table 2.

First, we can see that both BERT and BERT+SGM show favorable results on multi-label classification datasets mostly outperforming other baselines by a significant margin.

On *RCV1-v2* dataset, it is clear that the BERT-based models perform the best in micro-F_1 metrics. The methods dealing with the class structure (tree hierarchy in HMCN and HiLAP, label frequency in BERT+SGM) also have the highest macro-F_1 score.

In some cases, BERT performs better than the sequence-to-sequence version, which is especially evident on the *Reuters-21578* dataset. Since BERT+SGM has more learnable parameters, a possible reason might be a fewer number of samples provided on the dataset. However, sometimes

[2] https://github.com/lancopku/SGM

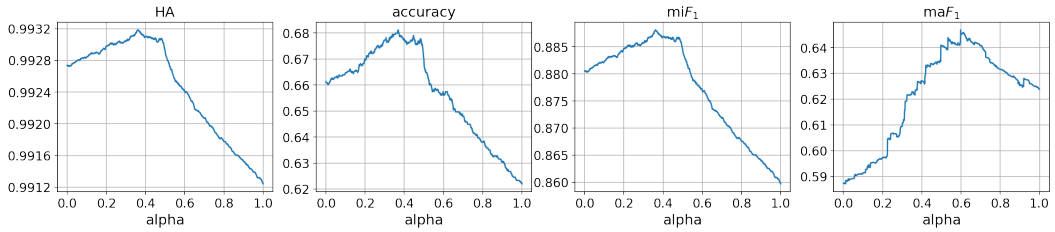

Figure 4: Hamming accuracy, set accuracy, miF_1, and maF_1 metrics of the mixed model on *RCV1-v2* validation set.

BERT+SGM might be a more preferable option: on *RCV1-v2* dataset the macro-F_1 metrics of BERT + SGM is much larger while other metrics are still comparable with the BERT's results. Also, for both Yandex Taxi datasets on the Russian language, we can see that the hamming accuracy and the set accuracy of the BERT+SGM model is higher compared to other models. On *Y.Taxi Riders* there is also an improvement in terms of macro-F_1 metrics.

In most cases, better performance can be achieved after mixing BERT and BERT+SGM. On public datasets, we see 0.4%, 0.8%, and 1.6% average improvement in miF_1, maF_1, and accuracy respectively in comparison with BERT. On datasets with tree hierarchy over classes, we observe 2.8% and 1.5% average improvement in maF_1 and accuracy. Metrics of interest for the mixed model depending on α on RCV1-v2 validation set are shown in Figure 4. Visualization of feature importance for BERT and sequence generating BERT models is provided in Appendix A.

In our experiments, we also found that BERT for multi-label text classification tasks takes far more epochs to converge compared to 3-4 epochs needed for multi-class datasets (Devlin et al., 2018). For *AAPD*, we performed 20 epochs of training; for *RCV1-v2* and *Reuters-21578* – around 30 epochs; for Russian datasets – 45-50 epochs. BERT + SGM achieves decent accuracy much faster than multi-label BERT and converges after 8-12 epochs. The behavior of performance of both models on the validation set of *Reuters-21578* during the training process is shown in Figure 3.

Another finding of our experiments is that the beam size in the inference stage does not appear to influence much on the performance. We obtained optimal results with the beam size in the range from 5 to 9. However, a greedy approach with the beam size 1 still gives similar results with less than 1.5% difference in the metrics. A possible explanation for this might be that, while in neural machine translation (NMT) the word ordering in the output sequence matters a lot and there might be confusing options, label set generation task is much simpler and we do not have any problems with ordering. Also, due to a quite limited 'vocabulary' size $|\mathcal{L}|$, we may not have as many options here to perform a beam search as in NMT or another natural sequence generation task.

5 Conclusion

In this research work, we examine BERT and sequence generating BERT on the multi-label setting. We experiment with both models and explore their particular properties for this task. We also introduce and examine experimentally a mixed model which is an ensemble of vanilla BERT and sequence-to-sequence BERT models.

Our experimental studies showed that BERT-based models and the mixed model, in particular, outperform current baselines by several metrics achieving state-of-the-art results on three well-studied multi-label classification datasets with English texts and two private Yandex Taxi datasets with Russian texts. We established that multi-label BERT typically needs several dozens of epochs to converge, unlike to BERT+SGM model which demonstrates decent results just after a few hundreds of iterations (less than a half of an epoch).

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012.

William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, pp. 307–315, New York, NY, USA, 1996. ACM. ISBN 0-89791-792-8. doi: 10.1145/243199.243278.

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

Sepp Hochreiter and Jrgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Jeremy Howard and Sebastian Ruder. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146, 2018.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1746–1751, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2015. Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5(Apr):361–397, 2004.

Junyang Lin, Qi Su, Pengcheng Yang, Shuming Ma, and Xu Sun. Semantic-unit-based dilated convolution for multi-label text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 4554–4564, 2018.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. Hierarchical text classification with reinforced label assignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, EMNLP 2019*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc., 2013.

Piero Molino, Huaixiu Zheng, and Yi-Chia Wang. COTA: improving the speed and accuracy of customer support through ranking and deep networks. *KDD*, 2018.

Jinseok Nam, Jungi Kim, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification - revisiting neural networks. *CoRR*, abs/1312.5419, 2013.

Jinseok Nam, Eneldo Loza Menc'ia, Hyunwoo J. Kim, and Johannes Fürnkranz. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5419–5429, 2017.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pp. 1063–1072, 2018. doi: 10.1145/3178876.3186005.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL*, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *Technical report. OpenAI*, 2018.

Ruslan Salakhutdinov, Antonio Torralba, and Joshua B. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, pp. 1481–1488. IEEE Computer Society, 2011. ISBN 978-1-4577-0394-2.

Amaia Salvador, Michal Drozdzal, Xavier Giró i Nieto, and Adriana Romero. Inverse cooking: Recipe generation from food images. *CoRR*, abs/1812.06164, 2018.

Nitish Srivastava and Ruslan Salakhutdinov. Discriminative transfer learning with tree-based priors. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 2094–2102, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3104–3112, 2014.

Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. CNN-RNN: A unified framework for multi-label image classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2285–2294, 2016. doi: 10.1109/CVPR.2016.251.

Jonatas Wehrmann, Ricardo Cerri, and Rodrigo C. Barros. Hierarchical multi-label classification networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 5225–5234, 2018.

Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. *CoRR*, abs/1606.02960, 2016.

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. HD-CNN: hierarchical deep convolutional neural networks for large scale visual recognition. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 2740–2748, 2015. doi: 10.1109/ICCV.2015.314.

Pengcheng Yang, Shuming Ma, Yi Zhang, Junyang Lin, Qi Su, and Xu Sun. A deep reinforced sequence-to-set model for multi-label text classification. *CoRR*, abs/1809.03118, 2018a.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. SGM: sequence generation model for multi-label classification. In *COLING*, pp. 3915–3926. Association for Computational Linguistics, 2018b.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1174.

A FEATURE IMPORTANCE IN BERT AND SEQUENCE GENERATING BERT

A natural question arises as to whether the success of the mixed model is the result of two models having different views on text features. To have a rough idea of how the networks make their prediction, we visualized the word importance scores for each model using the leave-one-out method in Figure 5. It can be seen from this example that BERT+SGM seems to be slightly more selective in terms of features to which it pays attention. Also, in this particular case, the predictions of sequence generating BERT are more accurate.

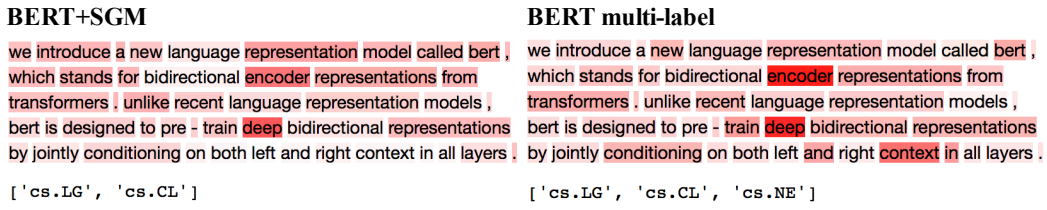

Figure 5: Visualization of feature importance for multi-label BERT and BERT+SGM models trained on *AAPD* and applied to BERT paper (Devlin et al., 2018) abstract (`cs.LG` – machine learning; `cs.CL` – computation & linguistics; `cs.NE` – neural and evolutionary computing).

B LABEL EMBEDDINGS IN MULTI-LABEL BERT

We extracted and projected to 2D-plane the label embeddings obtained from the fully connected classification layer of multi-label BERT fine-tuned on *AAPD* dataset. Visualization of some labels is shown in Figure 6. From this plot, we can see some clusters of labels that are close in terms of word.

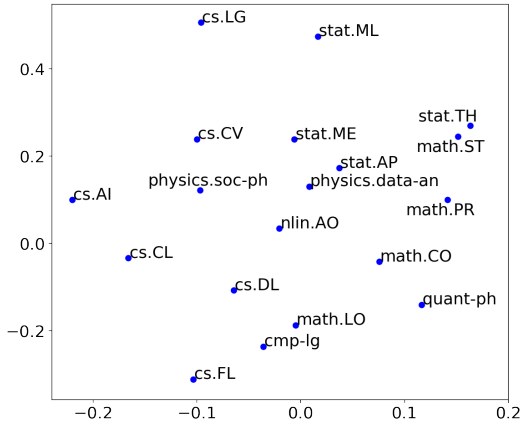

Figure 6: Projection of label embeddings obtained from the fully connected classification layer of multi-label BERT fine-tuned on *AAPD* dataset.