Deep Neural Models for ICD Coding from Clinical Text

Isabel Coutinho, Bruno Martins, João Leal

Abstract: The International Classification of Diseases (ICD) has been adopted worldwide in the healthcare domain. However, manual ICD coding of clinical documents is both time-consuming and error-prone, and it represents a huge monetary burden for a health facility. Thus, machine learning and deep learning algorithms can and have been used to automate ICD coding. This dissertation presents a novel deep neural network method for assigning ICD codes to clinical discharge summaries, combining word embeddings, recurrent units, and neural attention. The neural network explores the hierarchical nature of the input data by building representations at word and sentence-levels, also including at each level an attention mechanism. Moreover, several innovative strategies were tested together with the proposed model, including multi-label smoothing regularization, leveraging the hierarchical structure of the ICD codes, as well as data augmentation strategies or the use of alternative recurrent units. Experiments were conducted on the publicly available MIMIC III dataset, showing that the proposed model outperforms several previous deep learning models in most performance metrics. The proposed approach has the potential to be applied in hospitals and other health facilities, as part of a recommendation system for clinical coding.

Keywords: Automatic ICD Coding, Hospital Discharge Summaries, Deep Learning, Natural Language Processing, Multi-Label Classification

1 Introduction

In the last decade, we have seen the progressive adoption of the Electronic Health Record (EHR) worldwide, which consists of a computerized repository for patients' health information [1], resulting in a tremendous amount of digital health data. Typically, EHRs include structured data, such as laboratory results, medications, and diagnoses, but also unstructured text, such as radiology reports, progress notes, discharge summaries, and other clinical narratives. The use of all these data represents a promising approach to analyze patient information and better inform clinical decision support systems that can deliver personalized recommendations in real-time.

In this context, the World Health Organization (WHO) proposed the International Classification of Diseases (ICD)[1] coding system, which has been widely adopted by physicians and other health care providers, as this represents a standardized way of indicating diagnoses and procedures that are performed during a patient visit. ICD codes are used for many purposes, such as epidemiological studies, billing, and predictive modeling of the patient state [2]. Traditional ICD coding relies on human experience, which is time-consuming, error-prone, and represents a substantial monetary burden for a health facility. Medical coders review all pertinent medical record information and attribute the appropriate codes following rigid guidelines and conventions [2, 3]. Therefore, several types of errors frequently happen in this process. Firstly, ICD codes are organized in a hierarchical structure, where the top-level codes represent generic disease categories, and the bottom-level codes represent more specific diseases. It is common that medical coders either select incorrect subtypes of a particular disease, since the difference between disease subtypes is very subtle, or attribute an overly generic ICD code instead of a more specific one (*undercoding*). Also, clinical notes present abbreviations and synonyms, resulting in ambiguities and misunderstandings when coders are assigning ICD codes to the notes. Finally, there does not necessarily exist a one-to-one mapping between diagnosis descriptions and ICD codes. In many cases, several closely related diagnosis descriptions should be mapped to a single ICD code, resulting in an *unbundling* error if the physicians code each disease separately.

Given the aforementioned challenges, a huge effort has been placed on using machine learning and deep learning algorithms to automatically assign ICD codes to clinical text. However, this task still presents several challenges associated with the medical note representation and the medical coding system. Regarding the first case, clinical notes correspond to long text narratives with a vast medical vocabulary, making it difficult for a neural network model to encode and select critical information. Secondly, the medical coding system has a very high and sparse dimensional label space. There is a large number of available codes, e.g., over $17,000$ in ICD-9[2] and $180,000$ in ICD-10[3]. Also, most of the codes appear very seldom, resulting in few-shot learning problems, while a few codes occur substantially more than others. Finally, the automated models for ICD coding should be computationally efficient and generalizable across languages, avoiding the need for pretraining huge models on large amounts of texts from the clinical domain.

Methods for automatic ICD coding, using a supervised machine learning approach and specifically relying on deep neural networks, were already developed at Instituto Superior Técnico through a partnership with the Portuguese Directorate-General of Health (Direção Geral da Saúde), envisioning the coding of death certificates [4]. This work reports on extensions over that previous work, applying some novel ideas for automatic classification of full-text

[1] http://www.who.int/classifications/icd/

[2] https://www.cdc.gov/nchs/ICD/ICD9cm.htm

[3] https://www.cdc.gov/nchs/ICD/ICD10cm_pcs.htm

contents corresponding to hospital discharge summaries. Several innovative strategies were tested together with the proposed model, including multi-label smoothing regularization, as well as data augmentation strategies or the use of alternative recurrent units.

Following prior work, the proposed deep neural network method was evaluated on the publicly available Medical Information Mart for Intensive Care (MIMIC) III dataset [5]. Experimental results showed that the model with the best performance outperforms several previous deep learning models in most performance metrics. In addition, the proposed multi-label smoothing strategy, leveraging the hierarchical structure of the ICD codes, together with the adoption of a dice loss, in specific the log-cosh Tversky loss, in combination with the binary cross-entropy objective, proved to be very effective in this classification task. Data augmentation, based on a back-translation process, as well as pretraining the word embeddings with the domain data were also incorporated into the full proposed model.

The rest of the document is organized as follows. Section 2 outlines previous related work focusing on automatic ICD coding of clinical text. Section 3 presents the architecture of the deep neural network that was considered for addressing ICD coding as a supervised classification task, and the section also details all the extensions that were proposed on top of the previous model from Duarte et al. [4]. Section 4 reports the experimental evaluation, presenting dataset statistics for MIMIC III, the evaluation metrics, and the obtained results, establishing a comparison of different variants of the full model against state-of-the-art work in ICD coding. Finally, Section 5 summarizes the main conclusions and presents possibilities for future work in the area.

2 Related Work on Automatic ICD Coding

Automatic ICD coding has been a hot research problem in the clinical informatics domain for more than two decades [6, 7]. Early work for assigning ICD codes to clinical text usually relied on supervised machine learning approaches. Perotte et al. [8] described two approaches, both utilizing Support Vector Machines (SVMs): one treats each ICD code independently (flat SVM) and the other uses the hierarchical nature of the ICD codes (hierarchy-based SVM), showing that when the hierarchical nature of the codes is leveraged, the modeling is improved. Koopman et al. [9] proposed a hierarchical SVM approach to assign cancer-related ICD-10 codes to death certificates. First, a single binary classifier is trained to assign a cancer/nocancer label to a particular death certificate; then, if positive, a multiple classifier, one for each type of cancer, is used to assign a specific ICD-10 code to a death certificate. This two-level architecture lead to a substantial improvement in classification effectiveness.

More recently, due to the breakthroughs obtained by neural network models in several Natural Language Processing (NLP) problems [10], deep learning approaches have been proposed to handle the ICD coding task. Prakash et al. [11] introduced condensed memory neural networks (C-MemNNs), i.e., a model with iterative condensation of memory representations that preserves the hierarchy of features in the memory. The researchers combine an external clinical knowledge source (in this case, information from Wikipedia) with the free-text clinical notes, and use memory networks' learning capability to infer the most probable diagnosis correctly. This work's major contribution to memory networks is the addition of a condensed memory state, which is obtained via the iterative concatenation of successively lower-dimensional representations of the input memory state. Experiments showed that the proposed model outperforms other variants of memory networks to predict the most probable diagnoses.

Shi et al. [12] proposed a hierarchical deep learning model, featuring an attention mechanism, for automatically assigning ICD diagnosis codes given written diagnosis. For each diagnosis description, they use both character-level Long-Short Term Memory (LSTM) and word-level LSTM networks to obtain intermediate representations. They also employ a two-level LSTM architecture for each ICD code to obtain the hidden representation of its long title description. Since typically the number of written diagnosis descriptions does not equal the number of assigned ICD codes, they apply an attention mechanism for choosing which diagnosis descriptions are important when performing coding. This model outperforms others using character-unaware encoding methods or without attention mechanism.

A study conducted by Duarte et al. [4] addressed the assignment of ICD-10 codes for causes of death, by analyzing free-text descriptions, in death certificates, autopsy reports, and clinical bulletins from the Portuguese Ministry of Health. They leveraged a deep neural network that combines word embeddings, a hierarchical arrangement of recurrent units, neural attention, and mechanisms for initializing the weights of the final nodes of the network. The neural network explores the hierarchical nature of the input data, i.e., words from different fields (word-level) and the fields from different documents (field-level), using a bidirectional Gated Recurrent Unit (bi-GRU) and an attention mechanism at both levels. Furthermore, the authors proposed the initialization of the output nodes with the result of a Non-negative Matrix Factorization (NMF), applied to a matrix that encodes label co-occurrences in the training data. Experimental results attested to the contribution of the different neural network components, such as the attention mechanism and the NMF initialization. The model proposed by Duarte et al. [4] is, in fact, the main source of inspiration for the approach described in this work.

Mullenbach et al. [13] presented Convolutional Attention for Multi-Label classification (CAML), i.e., a Convolutional Neural Network (CNN) based method for automatic ICD code assignment. The authors employed a label-wise attention mechanism in ICD coding, which allows the model to learn distinct document representations for each label. The neural network passes text through a convolutional layer to compute a base representation of the text of each document making binary classification decisions for each ICD code. Rather than a pooling operation, they apply an attention mechanism to select the most relevant parts of the document for each possible code. Also, a regularization component encourages each code's parameters to be similar to those of codes with similar textual descriptions (DR-CAML). Although CAML outperforms previous models on all metrics, DR-CAML performs worse on most metrics when compared with CAML. The experiments were conducted on

the MIMIC datasets [5, 14], and the splits of datasets were publicly available, becoming a milestone for reproducibility in terms of methods for automated ICD coding.

However, models such as the one proposed by Mullenbach et al. [13] employ flat and fixed-length convolutional architectures. There are evident drawbacks in this type of approaches for multi-label clinical classification, which clearly requires variable-size features (such as texts fragments about diseases or procedures) for better representation, since the length and grammar vary significantly in different documents [15, 16]. Therefore, Xie et al. [15] improved the convolutional attention model by leveraging a densely connected CNN together with multi-scale feature attention. Their CNN consists of several stacked convolution blocks via dense connections, which can produce variable n-gram features layer per layer. After that, the authors apply a multi-scale feature attention mechanism to adaptively select the most informative n-gram features for each word according to the neighborhood. The authors also incorporate graph CNN to capture both hierarchical relationships among medical codes and the semantics of each code. The proposed model, named, MSATT-KG outperforms the CAML method by a considerable margin. Li and Yu [16] proposed a novel CNN architecture, combining multi-filter CNN and residual CNN. To capture patterns with different lengths, the authors leverage the multi-filter CNN, where each filter has a different window kernel size (i.e., word window size). On top of each filter, there is a residual convolutional layer, which consists of several residual blocks. Each of these blocks consists of three convolutional filters, allowing the enlargement of the receptive field. Experiments showed that MultiResCNN performs better than CAML in most evaluation metrics.

3 Proposed Approach

Taking as main inspiration the work developed by Duarte et al. [4], this work presents a novel deep neural network method for assigning ICD codes to clinical text, specifically by analyzing the free-text information within hospital discharge summaries. The network includes several tools to generate representations for textual contents, namely word embeddings, a hierarchical structure of the recurrent units, and neural attention. Moreover, several innovative strategies are proposed as extensions over the work from Duarte et al. [4], such as a multi-label smoothing regularization strategy, leveraging the hierarchical structure of the ICD codes. Mechanisms for addressing the data imbalance issue and for data augmentation were also incorporated. Finally, different mechanisms for initializing the weights of the final nodes of the network, and the use of alternative recurrent units are explored. Fig. 1 illustrates the proposed neural network architecture, which is detailed in the next sections.

Section 3.1 details the internal structure of the neural network architecture. Section 3.2 focuses on multi-label smoothing regularization, loss functions and data augmentation. Section 3.3 presents the different neural attention mechanisms that were explored. Section 3.4 describes two distinct approaches for initializing the weights of the final nodes. Section 3.5 concerns a recently proposed recurrent unit, namely the Mogrifier LSTM [17], which is also considered as an alternative recurrent unit.

3.1 A Neural Network Architecture for Automatic ICD Coding

Inspired on the proposal from Yang et al. [18], posteriorly extended by Duarte et al. [4], a two-level hierarchical approach is applied for modelling the input text. Therefore, the model starts by building representations of each sentence, taking word embeddings as input (word-level), which are aggregated into an encompassing representation. Then, those representations generated at the first level are used as input for the second level (sentence-level). GRUs can be used at both levels to build the representations, specifically considering bidirectional arrangements (e.g., bi-GRUs).

The GRU [19] computes a hidden state h_t based on the previous hidden state h_{t-1} and the current input x_t using two gates: a reset gate r_t and an update gate z_t, as shown in Eq. 1. These gates control how the information is updated. The update gate (Eq. 2) is used to determine how much past information is kept and how much new information is added, while the reset gated (Eq. 4) is used to control access to the previous state. In Eqs. 1-4, $\tilde{h}_t$ corresponds to the current new state, W and U are the parameter matrices for the actual and previous states, respectively, and b is a bias vector. The function α is the logistic sigmoid function, and $\odot$ is the elementwise product.

$$h_t = \mathrm{GRU}(x_t, h_{t-1}) = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (1)$$

$$z_t = \sigma(W_z \times x_t + U_z \times h_{t-1} + b_z) \quad (2)$$

$$\tilde{h}_t = \tanh(W_h \times x_t + r_t \odot (U_h \times h_{t-1} + b_h)) \quad (3)$$

$$r_t = \sigma(W_r \times x_t + U_r \times h_{t-1} + b_r) \quad (4)$$

A bi-GRU perceives the context of each input in a sequence by outlining the information from both directions (i.e., left to right and right to left). Therefore, the output at a position i is based on the concatenation of two output vectors produced by separate GRUs, taking into account both past and future, i.e., $h_{it} = [\overrightarrow{h_{it}}, \overleftarrow{h_{it}}]$.

The model also includes an attention mechanism at word and sentence-levels. This way, selective attention is paid to each word/sentence, i.e., different weights are used for the elements in the sequence of GRU outputs. At each level, the outputs h_{it} of the bi-GRU encoder are fed into a feed-forward node (Eq. 5), resulting in vectors u_{it} representing words or sentences in the input. The matrix W_w corresponds to the transformation matrix, and b_w to a bias term. Then, the attention weights α_{it} are computed according with Eq. 6, using a context vector u_w that is randomly initialized. These weights are summed over the whole sequence (Eq. 7). Finally, the vector s_i, which corresponds to a weighted sum of the bi-GRU outputs, is taken as the representation of the input of each level.

$$u_{it} = \tanh(W_w \times h_{it} + b_w) \quad (5)$$

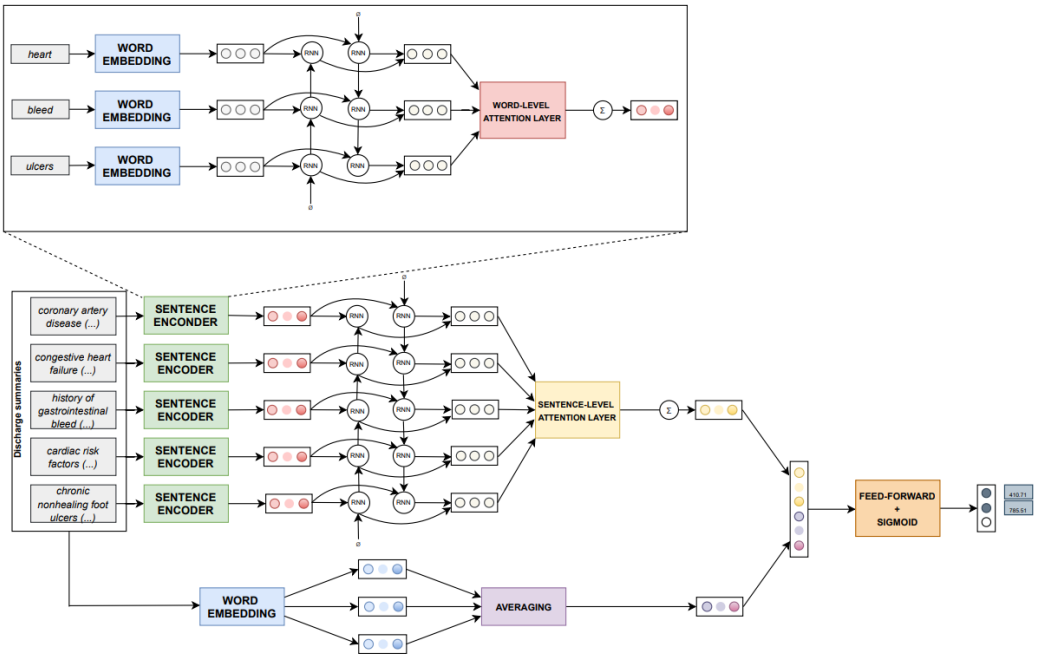

Fig. 1: The proposed neural network architecture.

$$\alpha_{it} = \frac{\exp(u_{it}^T \times u_w)}{\sum_t \exp(u_{it}^T \times u_w)} \tag{6}$$

$$s_i = \sum_t \alpha_{it} \times h_{it} \tag{7}$$

The output of the sentence-level attention layer is concatenated with the average of the embeddings for all words in the input field. The result of the concatenation is finally passed to a feed-forward output layer, with a number of nodes that is compatible with the classification task and with a sigmoid activation function.

3.2 Training Objective

Each discharge summary is usually associated with a set of ICD codes. Hence, the coding task is formulated as a multi-label classification problem. This work proposes multi-label smoothing regularization for better model calibration, i.e., aligning the confidence of the model's predictions with the accuracies of their predictions [20]. This strategy takes advantage of the hierarchical structure associated to ICD codes. Instead of considering a binary value for each ICD code, i.e., attributing $y_i = 1$ if the code is assigned and $y_i = 0$ otherwise, the ground-truth considers intermediate values for the codes belonging to the same blocks as the identified codes ($y_i = 0.05$). Paying attention to the codes of the same block may help the network to effectively distinguish the correct ICD codes together with these high similarity codes.

The binary cross-entropy (BCE) is used as the loss function, which is defined as follows:

$$\text{BCE} = \sum_i^m [-y_i \log(\tilde{y}_i) - (1 - y_i) \log(1 - \tilde{y}_i)] \tag{8}$$

In the previous expression, y_i and $\tilde{y}_i$ are, respectively, the ground-truth and the prediction for the i-th code. All parameters are learned by minimizing the loss function.

The use of a variant of a dice loss, in combination with the binary cross-entropy, is also explored in this work, to address the data-imbalance issue [21], promoting the correct prediction of the full set of ICD codes associated to each discharge summary. More specifically, it is considered the Tversky Loss (TL), an extension of the dice loss, which can be represented as follows:

$$\text{TL} = 1 - \frac{\sum_i^m y_i \tilde{y}_i + \gamma}{\sum_i^m y_i \tilde{y}_i + (1-\beta) \sum_i^m y_i (1-\tilde{y}_i) + \beta \sum_i^m (1-y_i) \tilde{y}_i + \gamma} \tag{9}$$

In the previous expression, β is chosen such that recall is considered β times as important as precision and γ is a factor added for smoothing purposes. Additionally, it is used a variant of the Tversky loss, namely log-cosh Tversky Loss (LCTL), which can be defined as:

$$\text{LCTL} = \log(\cosh(\text{TL})) = \log\left(\frac{e^{\text{TL}} + e^{-\text{TL}}}{2}\right) \tag{10}$$

With respect to data augmentation, two different strategies are explored. The first one consists of generating new discharge summaries by mixing the ones that present similar content. To do so, the first step is to choose two documents to be merged. Rather than randomly choosing the two documents, we first randomly choose one ICD code, and then two discharge summaries to which that specific code is assigned. This way, each code has the same probability of being chosen, and it is least probable to aggravate the data imbalance issue. Also, since the two discharge summaries present at least a particular ICD code in common, one assures that the content is related. After the concatenation of the two texts, a simple strategy to mix the resulting content

is performed: a small number of sentences (between two and four) is selected and randomly introduced in another position of the document. The ICD codes of the two instances are also concatenated, resulting in a new training instance.

The other strategy corresponds to back-translation. This process is achieved by using Google translate to convert an original document into a new language, and then taking the translated document and translate it back into the original language. Thus, a new document is generated since we do not achieve exactly the same original text, resulting in text with a different structure but preserving the semantic content. This allows the model to pay more attention to content rather than accessory information (e.g., pronouns or conjunctions). In this case, for each discharge summary, five different new documents are obtained, corresponding to five different languages in which the back-translation process is performed (Portuguese, Spanish, Italian, French, and German). The implementation of this strategy relied on googletrans[4]. For each discharge summary, the least similar generated document (considering Jaccard similarity) is chosen to obtain a dataset as varied as possible. Also, new document words that are not included in the word vocabulary from the original training set are substituted by the most similar word, according to the Jaro-Winkler string distance metric [22].

3.3 Neural Attention Mechanisms

Regarding the neural attention mechanisms, this work proposes three variants of the approach described by Duarte et al. [4] and represented in Eqs. 5-7. Focusing on Eq. 5, one can add a term with a value obtained from the operation of max pooling over the output h_{it} of the bi-GRU encoder, which would be given from:

$$m_t = \arg\max_j h_{jt}, \tag{11}$$

The max pooling operation is used to process the vectors resulting from the sequence into a single d-dimensional vector by taking the max over each dimension [10]. The pooling operation can be seen as a feature extractor that extracts the most important features, where the order of the features is preserved but not their specific positions. In this context, this means that at the word and the sentence-levels, the most salient information across the word and the sentence sets, respectively, is selected. Including this term when computing the vectors u_{it}, representing words or sentences for the attention mechanism as in Eq. 5, one obtains:

$$u_{it} = \tanh(W_w \times h_{it}) + (W_m \times m_t) + b_w \tag{12}$$

In the previous expression, W_m is the transformation matrix associated with max pooling term.

Another possible modification to the simper attention mechanism is the addition of a term representing the average of the embeddings for all words in the entire input. Notice that this differs from an average pooling operation, in the sense that we are using embeddings for the original words, instead of the vectors produced by the Recurrent Neural Network (RNN). Given a clinical note with N words, denoted as $X = \{x_1, x_2, ..., x_N\}$, and the word embedding matrix denoted as $E = [e_1, ..., e_N]^T \in \mathbb{R}^{N \times d_e}$, where d_e is the dimension of word embeddings, the average of word embeddings is represented as follows:

$$a = \frac{1}{N} \sum_{i=1}^{N} e_i \tag{13}$$

Considering this term for the computation of u_{it}, one obtains:

$$u_{it} = \tanh(W_w \times h_{it}) + (W_a \times a) + b_w \tag{14}$$

In the previous expression, W_a is the transformation matrix associated with the average of embeddings term.

Including both terms in the attention mechanism, one obtain the following alternative to Eq. 5:

$$u_{it} = \tanh(W_w \times h_{it}) + (W_m \times m_t) + (W_a \times a) + b_w \tag{15}$$

3.4 Initializing the Weights of the Output Nodes

Regarding the initialization of the final nodes in the neural network, two different methodologies were exploited. As suggested by Duarte et al. [4], the first technique is based on a NMF over a label co-occurrence matrix. First, a square matrix $X_{m,m}$ is built, being m the dimensionality of the output layer in the model. Each matrix cell reflects the number of co-occurrences of a pair of ICD labels in the entire training set (the diagonal corresponds to the frequency of the label). In order to reduce the impact of the most common labels, each entry $x_{i,j}$ of the matrix $X_{m,m}$ is scaled with a binary logarithm, $\log_2(1 + x_{i,j})$. Then, the NMF is used to decomposed the $X_{m,m}$ matrix into a product of two matrices as $X_{m,m} \approx W_{m,n} \times H_{n,m}$, where n is the dimensionality of the output of the node before the final node. The matrix $H_{n,m}$ is used as the initialization.

The other methodology for initializing the weights of the final nodes also leverages the co-occurrences between ICD codes, taking pretrained label embeddings using the GloVe model [23] on the label set of the training data.

3.5 Using Mogrifier LSTMs as the Recurrent Units

Alternatively to GRUs, other recurrent units can be used at both levels of the neural network, namely LSTMs [24]. There are three gates, i_t, f_t and o_t, controlling for input, forget and output, respectively. The gate values are computed based on linear combinations of the current input x_t and the previous state h_{t-1}, passed through a sigmoid activation function (Eqs. 18-20). An update candidate g_t (Eq. 21) is computed as a linear combination of x_t and h_{t-1}, passed through an hyperbolic tangent activation function. The memory component c_t (Eq. 17) is then updated: the forget gate f_t controls how much of the previous memory to keep, and the input gate i_t controls how much of the proposed update to keep. Finally, the output h_t (Eq. 16) is determined based on the content of the memory c_t passed through an hyperbolic tangent nonlinearity and controlled by the output gate o_t. In Eqs. 18-21, W and U are the parameter matrices for the actual and previous states, respectively, and b is a bias vector.

[4]https://pypi.org/project/googletrans/

$$h_t = \text{LSTM}(x_t, c_t, h_{t-1}) = o_t \odot \tanh(c_t) \quad (16)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (17)$$

$$i_t = \sigma(W_i \times x_t + U_i \times h_{t-1} + b_i) \quad (18)$$

$$f_t = \sigma(W_f \times x_t + U_f \times h_{t-1} + b_f) \quad (19)$$

$$o_t = \sigma(W_o \times x_t + U_o \times h_{t-1} + b_o) \quad (20)$$

$$g_t = \tanh(W_g \times x_t + U_g \times h_{t-1} + b_g) \quad (21)$$

In this work, we considered an extension to the original LSTM architecture, namely the Mogrifier LSTM [17]. Mogrifier LSTM is a LSTM where two inputs x_t and h_{t-1} modulate one another in an alternating fashion before the usual LSTM computation takes place, as follows:

$$\text{MofigierLSTM}(x_t, c_t, h_{t-1}) = \text{LSTM}(x_t^{\uparrow}, c_t, h_{t-1}^{\uparrow}) \quad (22)$$

In the previous expression, the modulated $x_t^{\uparrow}$ and $h_{t-1}^{\uparrow}$ are defined as the highest indexed x_t^i and h_{t-1}^i, respectively, from the interleaved sequences:

$$x_t^i = 2\sigma(Q^i h_{t-1}^{i-1}) \odot x_t^{i-2}, \qquad \text{for odd } i \in [1, ..., r] \quad (23)$$

$$h_{t-1}^i = 2\sigma(R^i x_t^{i-1}) \odot h_{t-1}^{i-2}, \qquad \text{for even } i \in [1, ..., r] \quad (24)$$

with $x_t^{-1} = x_t$ and $h_{t-1}^0 = h_{t-1}$. The number of rounds, $r \in \mathbb{N}$, is an hyperparameter. To reduce the number of additional model parameters, the matrices Q^i and R^i can be factorized as products of low-rank matrices: $Q^i = Q_{left}^i Q_{right}^i$ with $Q^i \in \mathbb{R}^{m \times n}$, $Q_{left}^i \in \mathbb{R}^{m \times k}$, $Q_{right}^i \in \mathbb{R}^{k \times n}$, where $k < \min(m, n)$ is the rank.

4 Experimental evaluation

This section describes the experimental evaluation of the proposed method, presenting a statistical characterization of the dataset, the evaluation metrics, and the obtained results establishing a comparison between different variants of the proposed model with the best performance and state-of-the-art-work on ICD coding.

4.1 Dataset

Following the line of work presented by Mullenbach et al. [13], the experiments were conducted using the publicly available MIMIC III dataset [5], which comprises information relating to patients admitted to critical care units. We specifically used the exact same data splits from the work of Mullenbach et al. [13], which were also used in many of the subsequent studies in the area. Hospital discharge summaries were the clinical text considered, as these condense all the information during a patient visit into one document. In the case of the admissions having addenda to their summary, these were concatenated to form a single document.

TABLE 1: Descriptive statistics for MIMIC III dataset.

Number of distinct ICD-9-CM codes	8,921
Number of distinct ICD-9-CM blocks	1,158
Number of distinct ICD-9-CM chapters	32
Average number of ICD-9-CM codes per instance	15.9
Average number of ICD-9-CM blocks per instance	13.8
Average number of ICD-9-CM chapters per instance	7.7
Number of discharge summaries in the dataset	52,722
Average number of words per instance	1,513.5
Average number of sentences per instance	109.3
Average number of words per sentence	13.8
Training set vocabulary size	139,096
Number of OOV words in the validation set	3,446
Number of OOV words in the test set	6,579

Each admission is tagged by human coders with a set of ICD-9-CM codes, describing both diagnoses and procedures during the patient visit. There are $8,921$ unique ICD codes present in the dataset, including $6,918$ diagnosis codes and $2,003$ procedure codes. The data was split using the patient ID so that no patient appears on both training and test sets. Patient visits that contain no assigned diagnoses or procedures were discarded. Table 1 details the dataset statistics, considering the three data splits (i.e., training, validation and test).

Different experiments were conducted on MIMIC III, using the full-label setting as well as the top-50 most frequent codes, designated from now on as MIMIC-III-full and MIMIC-III-50, respectively. The MIMIC-III-full experiments considered a total of $52,722$ discharge summaries, namely $47,719$ for training, $1,631$ for validation and $3,372$ for testing. MIMIC-III-50 corresponds to a subset of $11,368$ discharge summaries, in which $8,066$, $1,573$ and $1,729$ are used for training, validation and test sets, respectively.

When preprocessing the data, similarly to previous works in the area [13, 15, 16], tokens containing no alphabetic characters were removed and all tokens were converted to lower case. As described previously, the model takes a hierarchical structure, building a representation of each sentence firstly, and then of the sequence of sentences. Therefore, besides tokenization of the input texts, one more level of segmentation was carried out, at the sentence-level, using the spaCy library[5]. The number of sentences and the number of words of each sentence were truncated to the percentile 95 of each parameter over the training set (i.e., 211 and 27, respectively), reducing the computational cost. Special tokens determine the beginning and end of each sentence.

The word vocabulary was generated using only the training instances, having a vocabulary size of $139,096$ and $58,452$ tokens for MIMIC-III-full and MIMIC-III-50, respectively. Out-Of-Vocabulary (OOV) words included in the validation and test sets were substituted by the most similar word on the vocabulary, according to the Jaro-Winkler string distance metric [22]. As described by Duarte et al. [4] and confirmed through manual analysis, this approach reduces the number of misspellings or alternative spellings for a word, by substituting it for the most similar one.

[5] https://spacy.io

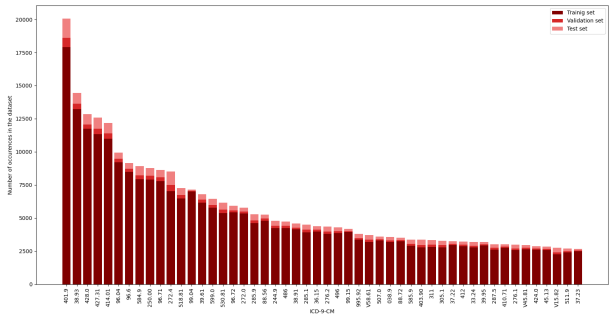

Fig. 2: Number of occurrences of the 50 most common ICD-9-CM codes in the dataset.

One of the main challenges in the ICD coding classification task relates to the label distribution being extremely imbalanced. Most of the codes appear very seldom, while few codes occur several orders of magnitude more than others. Note that $5,233$ possible codes occur less than 10 times in the dataset. Fig. 2 shows the distribution for the 50 most common ICD-9-CM codes in the dataset.

Another challenge concerning the multi-label scenario is the high number of ICD codes assigned to each discharge summary. Although the average number of unique labels per instance is 15.9, it is possible to find up to 71 codes associated with one discharge summary.

4.2 Evaluation Metrics

To establish a fair comparison with prior work, the results of the proposed model are reported on a variety of metrics, focusing on micro-averaged and macro-averaged F1, AUC (Area Under the Curve), and precision at n [13]. Micro-averaged values treat each pair text-code as a separated prediction. On the other hand, macro-averaged values are computed by averaging metrics computed per-label. Precision at n, denoted as P@n, corresponds to the precision of the n highest scoring labels, checking if they are present in the ground truth. We have chosen $n = 5$ for evaluating the experiments conducted on MIMIC-III-50, which roughly corresponds to the average number of codes, and $n = 8$ and $n = 15$ for the experiments with MIMIC-III-full.

4.3 Experiments and Results

The implementation of the model relied mostly on TensorFlow[6] deep learning library. Other machine learning and NLP libraries such as scikit-learn[7] or NLTK[8] were also used for specific tasks. The entire model was trained using the backpropagation algorithm [25] in conjunction with the Adaptive Moment Estimation (Adam) optimization method [26], setting its learning rate to the default value of 0.001. The word embedding layer in the first level of the neural model and the output dimensionality of the RNNs were set to 175. The batch size and the maximum number of epochs were defined as 32 and 50, respectively. An early stopping mechanism was also used, in which the training was stopped if there was no improvement in the loss over the validation set, in two continuous epochs.

In order to assess the importance of each proposed feature, different variants of the full model outlined in Section 3 were evaluated. These alternatives are presented in this section. Starting from a base model, the following features were tested: (i) including the multi-label smoothing regularization; (ii) combining the binary cross-entropy with the log-cosh Tversky loss; (iii) substituting the regular attention mechanism by other additive attention mechanisms; (iv) leveraging the label co-occurrences when initializing the weights of the output nodes; (v) substituting the GRU by alternative RNN cells, namely the Mogrifier LSTM [17]; (vi) incorporating data augmentation; and (vii) including pretrained word embeddings. Table 2 presents the results for the following distinct models:

1) **Base Model (BM)**: the architecture from Duarte et al. [4], without considering mechanisms for initializing the weights of the output nodes, and with randomly initialized word embeddings;
2) **BM + MLS**: a model like the previous one, including the proposed multi-label smoothing regularization method;
3) **BM + MLS + LCTL – Best Loss Model (BLM)**: the model described above, combining the binary cross-entropy with the log-cosh Tversky loss;
4) **BLM with attention as in Eq. 12**: Best Loss Model, including the max pooling term when computing the attention weights as in Eq. 12;
5) **BLM with attention as in Eq. 14**: Best Loss Model, including the average of embeddings term when computing the attention weights as in Eq. 14;
6) **BLM with attention as in Eq. 15**: Best Loss Model, including both max pooling and average of embeddings terms when computing the attention weights as in Eq. 15;
7) **BLM + NMF**: Best Loss Model, incorporating the initialization of the output nodes with basis on the NMF strategy;
8) **BLM + GloVe label embeddings**: similar to the previous one, but considering pretrained label embeddings resulting from the GloVe model [23] for initializing the output nodes;
9) **BLM with Mogrifier LSTM**: Best Loss Model, substituting the GRU by the Mogrifier LSTM units;
10) **BLM + augmentation**: Best Loss Model, including data augmentation performed through the concatenation of similar discharge summaries;
11) **BLM + back-translation**: similar to the latter, but using data augmentation through the back-translation strategy;
12) **BLM + back-translation + GloVe word embeddings**: Best Loss Model, including not only the back-translation mechanism, but also pretrained word embeddings using the GloVe method for initializing the embedding layer.

On what regards the contribution of the multi-label smoothing regularization, focusing on F1 metrics, one can verify that this strategy results in an increase of performance

[6]http://tensorflow.org
[7]http://scikit-learn.org
[8]http://www.nltk.org

TABLE 2: Performance metrics for comparison of different models on the MIMIC-III-full dataset. Here and in all the following tables, "Diag" and "Proc" denotes the micro-F1 performance on diagnosis and procedure codes only, respectively. The values in bold represent the best result in each metric.

Model	AUC		F1				P@n	
	Macro	Micro	Macro	Micro	Diag	Proc	8	15
BM	86.2	**97.8**	2.3	34.5	29.8	49.1	57.2	43.3
BM + MLS	**87.0**	96.9	2.6	34.6	30.9	47.0	56.8	43.2
BM + MLS + LCTL (BLM)	83.1	95.3	2.7	41.6	37.8	53.4	59.2	44.1
BLM with attention as in Eq. 12	83.1	95.4	2.3	40.5	36.8	52.3	58.1	41.2
BLM with attention as in Eq. 14	83.6	95.7	2.7	41.8	38.2	53.4	59.3	43.8
BLM with attention as in Eq. 15	83.2	95.4	2.7	41.9	38.5	53.2	59.5	42.5
BLM + NMF	84.5	95.5	2.5	41.6	38.4	52.3	59.9	42.8
BLM + GloVe label embeddings	81.5	95.4	2.0	37.3	34.0	48.1	54.7	40.8
BLM with Mogrifier LSTM	81.6	95.5	1.6	33.7	30.2	44.9	48.1	36.3
BLM + augmentation	82.8	95.3	2.6	40.6	36.9	52.5	58.0	43.2
BLM + back-translation	84.2	95.6	3.0	43.5	40.1	54.6	61.1	45.6
BLM + back-trans. + GloVe word emb.	84.5	95.9	**3.3**	**44.7**	**41.3**	**55.8**	**63.2**	**47.0**

(BM + MLS). As already mentioned, this strategy proposes that codes belonging to the same blocks as the assigned codes take the value $y_i = 0.05$ (rather than $y_i = 0$). Several tests were done to optimize this value, considering that more codes of the same block would be predicted if this value was higher, resulting in a higher recall. However, the precision would decrease, leading to a lower F1 score. Therefore, the value was chosen according to the F1.

Comparing BM + MLS with BM + MLS + LCTL – Best Loss Model (BLM), it is possible to verify that the combination of the binary cross-entropy with the log-cosh Tversky loss represents a significant performance increase, with better results in F1 (macro and micro-averaged) and P@n metrics. The weights considered in the loss were 0.9 and 0.1 for the binary cross-entropy and Tversky loss, respectively, and the hyperparameters $\beta = 0.5$ and $\gamma = 1$. These values were also chosen through a initial set of tests.

After defining the loss function, the following tests were done in parallel in order to evaluate the following features over the Best Loss Model: different attention mechanisms, initialization of the weights of the final nodes, substituting GRUs by Mogrifier LSTMs, and data augmentation.

Analyzing the results obtained when incorporating alternative attention mechanisms, the respective models produce a performance increase, reflected in several metrics. However, this increase is neither systematic nor very significant, so the use of these attention mechanisms was deemed as not compensatory.

Contrary to what was described by Duarte et al. [4], when considering a multi-label smoothing mechanism that leverages the hierarchical structure of the ICD codes and including the Tversky loss function, the NMF initialization of the final nodes does not produce better results. Similarly, the initialization with the pretrained label embeddings resulting from the GloVe model represents a worse performance. In this case, different experiments were also conducted in order to optimize the number training epochs hyperparameter (finally defined as five). Notice also that the resulting label embeddings were normalized to have only positive values, similarly to the NMF strategy.

Regarding the use of an alternative recurrent unit, a significant decrease of performance is obtained in all metrics when substituting the GRU by the Mogrifier LSTM. The optimal setting described in the original proposal for Mogrifier LSTMs includes setting the number of rounds $r \in \{5, 6\}$ and the rank $k \in [40, ..., 90]$ [17]. According to this, we considered $r = 5$ and $k = 65$.

With respect to data augmentation, BLM + augmentation includes new discharge summaries resulting from concatenating similar ones from the MIMIC dataset. For each batch of 32 instances, 8 correspond to new instances. However, this model produces a decrease in all evaluation metrics. One possible explanation is that although the concatenated discharge summaries have at least one ICD code in common, the content is not as similar as originally envisioned. Therefore, the combination results in adding noisy data to the original dataset. On the other hand, when duplicating the training set through the back-translation strategy (BLM + back-translation), one verifies a significant increase in all evaluation metrics. This process assumes the addition of real discharge summaries to the dataset, but that have a slightly different format. Thereby, this allows the model to pay more attention to content rather than accessory information.

Finally, the BLM + back-translation + GloVe word embeddings model corresponds to a combination of the best ideas and is the one with generally better results, with a significant performance increase reflected on all metrics, except AUC. In this case, several experiments were also taken to optimize the number of training epochs of the GloVe model, which was determined as 10.

4.4 Comparison with Previous Work

Regarding the MIMIC-III-full experiment, Table 3 shows the comparison of the proposed work against several previous works. Specifically, using a label-wise attention mechanism, CAML [13] produces better performance than previous deep learning models, based on standard CNNs [27] and bi-GRUs [19]. However, DR-CAML [13], which includes a description regularizer, performs worse on most metrics than CAML. Addressing the fixed window size problem of CAML, MASATT-KG [15] and MultiResCNN [16] achieve better results than CAML. The model with the best performance proposed in this work, BLM + back-translation + Glove

TABLE 3: Performance metrics for comparison with previous work on the MIMIC-III-full dataset.

	AUC		F1				P@n	
Model	Macro	Micro	Macro	Micro	Diag	Proc	8	15
CNN [27]	80.6	96.9	4.2	41.9	40.2	49.1	58.1	44.3
Bi-GRU [19]	82.2	97.1	3.8	41.7	39.3	51.4	58.5	44.5
CAML [13]	89.5	98.6	8.8	53.9	52.4	60.9	70.9	56.1
DR-CAML [13]	89.7	98.5	8.6	52.9	51.5	59.5	69.0	54.8
MSATT-KG [15]	91.0	99.2	9.0	55.3	–	–	72.8	58.1
MultiResCNN [16]	91.0	98.6	8.5	55.2	–	–	73.4	58.4
BLM + back-trans. + GloVe word emb.	84.5	95.9	3.3	44.7	41.3	55.8	63.2	47.0

TABLE 4: Performance metrics for comparison with previous work on the MIIMC-III-50 dataset.

	AUC		F1		
Model	Macro	Micro	Macro	Micro	P@5
C-Mem-NN [11]	83.3	–	–	–	42.0
C-LSTM-Att [12]	–	90.0	–	53.2	–
CNN [27]	87.6	90.7	57.6	62.5	62.0
Bi-GRU [19]	82.8	86.8	48.4	54.9	59.1
LEAM [28]	88.1	91.2	54.0	61.9	61.2
CAML [13]	87.5	90.9	53.2	61.4	60.9
DR-CAML [13]	88.4	91.6	57.6	63.3	61.8
MSATT-KG [15]	91.4	93.6	63.8	68.4	64.4
MultiResCNN [16]	89.9	92.8	60.6	67.0	64.1
BLM + back-trans. + GloVe word emb.	85.2	88.7	48.2	55.1	56.4

word embeddings, when compared to standard CNNs and bi-GRUs, produces higher results in the macro-AUC, micro-F1, and P@n metrics, while achieving lower micro-AUC and macro-F1. In particular, this model improves the macro-AUC by 2.3%, micro-F1 by 3.0%, P@8 by 4.7%, and P@15 by 2.5% comparing to the used of standard bi-GRU.

In turn, Table 4 shows the results on the MIMIC-III-50 dataset. The proposed model, compared to C-Mem-NN [11], produces notable improvements of 1.9% and 14.4% in macro-AUC and P@5, respectively. When compared to C-LSTM-Att [12], it also produces an improvement of 1.9% in micro-F1. Finally, comparing with a standard bi-GRU, it achieves a improvement of 2.4%, 1.9% and 0.2% in macro-AUC, micro-AUC and micro-F1, respectively.

5 Conclusions and Future Work

This work presented a deep learning method for automatically assigning ICD codes to clinical text, using hospital discharge summaries as a case study. Experiments conducted on the benchmark MIMIC III dataset [5] showed that the model with the best performance proposed in this work outperforms several deep learning models in most performance metrics. Furthermore, experimental results with different variants of the full model attest to the contribution of distinct model features, namely the proposed multi-label smoothing regularization, the use of the log-cosh Tversky loss together with the binary cross-entropy, data augmentation resulting from the back-translation mechanism, and pretrained word embeddings using the GloVe model [23]. We argue that this model has the potential to be integrated into existing coding platforms as a recommendation system, reducing the involvement of medical coders in the ICD coding.

The proposed model does not present any constraints or specifications regarding the classification system used or the clinical text taken as input, and can be used in any similar classification task. Therefore, considering the application of this work to real data from a Portuguese hospital rather than on a database such as MIMIC III, only a few adaptions would have to be made. The language difference would not be an obstacle since the model does not use external knowledge (e.g., large external datasets for pretraining word embeddings). Even though the model is very flexible, some limitations would be faced if a dataset different from MIMIC was used. For instance, the codes included in the MIMIC III dataset correspond to ICD-9-CM. The Ninth Revision differs significantly from the Tenth Revision, which is much more specific and comprises about 10 times more codes, resulting in a more large and sparse label set and making the coding task more difficult.

In order to improve the model performance, many options can be taken into account for future work. For instance, one major difficulty when developing deep learning methods for large-scale multi-label text classification problems, particularly for automatic ICD coding, is predicting infrequent or unseen labels. Thus, the investigation of few-shot or zero-shot learning problems is crucial. Recent work has already addressed this issue, to some degree, focusing on the ICD code descriptions and exploring the hierarchical structure [29, 30].

Acknowledgments

I would like to express my gratitude to INESC-ID and CUF for supporting this project. A special thanks to Prof. Bruno Martins and Eng. João Leal for their guidance.

REFERENCES

[1] I. Mosby. *Mosby's medical dictionary*. 2006.

[2] K. J. O'Malley, K. F. Cook, M. D. Price, K. R. Wildes, J. F. Hurdle, and C. M. Ashton. Measuring diagnoses: ICD code accuracy. *Health Services Research*, 40(5 II): 1620–1639, 2005.

[3] A. N. Nguyen, D. Truran, M. Kemp, B. Koopman, D. Conlan, J. O'Dwyer, M. Zhang, S. Karimi, H. Hassanzadeh, M. J. Lawley, and D. Green. Computer-Assisted Diagnostic Coding: Effectiveness of an NLP-based approach using SNOMED CT to ICD-10 mappings. In *Proceedings of American Medical Informatics Association Annual Symposium*, pages 807–816, 2018.

[4] F. Duarte, B. Martins, C. Sousa, and M. J. Silva. Deep neural models for ICD-10 coding of death certificates and autopsy reports in free-text. *Journal of Biomedical Informatics*, 80:64–77, 2018.

[5] A. E. Johnson, T. J. Pollard, L. Shen, L. W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3:1–9, 2016.

[6] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In *Proceeding of Special Interest Group on Information Retrieval*, pages 289–297, 1996.

[7] L. R. S. de Lima, A. H. F. Laender, and B. A. Ribeiro-Neto. A hierarchical approach to the automatic categorization of medical documents. In *Proceedings of Conference on Information and Knowledge Management*, pages 132–139, 1998.

[8] A. Perotte, R. Pivovarov, K. Natarajan, N. Weiskopf, F. Wood, and N. Elhadad. Diagnosis code assignment: Models and evaluation metrics. *Journal of the American Medical Informatics Association*, 21(2):231–237, 2014.

[9] B. Koopman, G. Zuccon, A. Nguyen, A. Bergheim, and N. Grayson. Automatic ICD-10 classification of cancers from free-text death certificates. *International Journal of Medical Informatics*, 84(11):956–965, 2015.

[10] Y. Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57(1):345–420, 2016.

[11] A. Prakash, S. Zhao, S. A. Hasan, V. V. Datla, K. Lee, A. Qadir, J. Liu, and O. Farri. Condensed Memory Networks for Clinical Diagnostic Inferencing. In *Association for the Advancement of Artificial Intelligence*, pages 3274–3280, 2017.

[12] H. Shi, P. Xie, Z. Hu, M. Zhang, and E. P. Xing. Towards Automated ICD Coding Using Deep Learning. *arXiv preprint arXiv:1711.04075*, 2017.

[13] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1101–1111, 2018.

[14] J. Lee, D. J. Scott, M. Villarroel, G. D. Clifford, M. Saeed, and R. G. Mark. Open-access MIMIC-II database for intensive care research. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pages 8315–8318, 2011.

[15] X. Xie, , Y. Xiong, P. S. Yu, and Y. Zhu. EHR Coding with Multi-scale Feature Attention and Structured Knowledge Graph Propagation. In *Proceeding of Conference on Information and Knowledge Management*, pages 649–658, 2019.

[16] F. Li and H. Yu. ICD Coding from Clinical Text Using Multi-Filter Residual Convolutional Neural Network. In *Proceedings of Association for the Advancement of Artificial Intelligence*, pages 8180–8187, 2020.

[17] G. Melis, T. Kočiský, and P. Blunsomy. Mogrifier LSTM. In *Proceedings of International Conference on Learning Representations*, pages 1–13, 2020.

[18] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1480–1489, 2016.

[19] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of the Workshop on Synthax, Semantics and Structure in Statistical Translation*, 2014.

[20] R. Müller, S. Kornblith, and G. Hinton. When does label smoothing help? In *Proceedings of Neural Information Processing Systems*, pages 1321–1330, 2019.

[21] X. Li, X. Sun, Y. Meng, J. Liang, F. Wu, and J. Li. Dice Loss for Data-imbalanced NLP Tasks. *arXiv preprint arXiv:1911.02855v2*, 2020.

[22] W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Census Bureau, RR99/04, 1999.

[23] J. Pennington, R. Socher, and C. D. Manning. GloVe : Global Vectors for Word Representation. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[24] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive Modeling*, 5(3):533–536, 1986.

[26] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference for Learning Representations*, 2015.

[27] Y. Kim. Convolutional Neural Networks for Sentence Classification. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 1746–1751, 2014.

[28] G. Wang, C. Li, W. Wang, Y. Zhang, D. Shen, X. Zhang, R. Henao, and L. Carin. Joint embedding of words and labels for text classification. In *Proceedings of ACL*, pages 2321–2331, 2018.

[29] C. Song, S. Zhang, N. Sadoughi, P. Xie, and E. Xing. Generalized Zero-shot ICD Coding. *arXiv preprint arXiv:1909.13154*, 2019.

[30] A. Rios and R. Kavuluru. Few-shot and zero-shot multi-label learning for structured label spaces. pages 3132–3142, 2020.