2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)
Bari, Italy. October 6-9, 2019

A Deep Learning Model with Data Enrichment for Intent Detection and Slot Filling

Sławomir Dadas, Jarosław Protasiewicz, and Witold Pedrycz
National Information Processing Institute, Warsaw, Poland

***Abstract*— A model for accurately solving the tasks of intent detection and slot filling requires a substantial amount of user queries annotated manually, which is time-consuming and costly. In this study, in order to circumvent this problem, we propose a method of expanding a training set for handling these two tasks. The data are augmented by applying a random mutation to the training samples, following a set of heuristic rules. For validation of the method, we construct a compound neural architecture composed of a long short-term memory layer, an attention mechanism, and conditional random fields. The experiments conducted on the Automatic Terminal Information Service (ATIS) dataset demonstrate that the models trained on the expanded datasets improve their F-score for the slot-filling task. This improvement is particularly significant for small datasets. We believe that this method allows for the expansion of a small set of previously annotated user queries into a large training set that is sufficient for correctly constructing a model of intent detection and slot filling. Therefore, our approach can be used to save time and money by reducing the amount of data required to train a natural language understanding model. Its novelty consists in expanding a training set for a deep learning model based on a random mutation of training samples, following a set of heuristic rules that utilise external lexicons and the training data itself.**

I. Introduction

Natural language understanding (NLU) is a sub-discipline of artificial intelligence which concerns the extraction and representation of the meanings of speech or text expressed in natural language. Methods aimed to solve this problem are useful in numerous practical applications, including human-computer interfaces, call centre automation (IVR systems), chat bots, and semantic search engines. A possible approach to language understanding is to reduce this problem to two straightforward tasks. The first one identifies the user intent, expressed as a short fragment of natural language text; the second one extracts additional contextual information (slots) that are relevant to the user intention.

Early approaches to intent detection and slot filling problems employed separate models for these two tasks. The common approaches were generally combinations of heuristics and statistical models with conditional random fields (CRF) and maximum entropy Markov models [1], [2], [3]. In recent years, a number of methods based on deep learning have been proposed. Modern methods mostly utilise recurrent neural networks [4], [5], [6] or convolutional neural networks [7]. The flexibility of neural architectures allowed for the development of joint algorithms, in which a single model is optimised to perform intent detection and slot filling simultaneously, usually leading to improved performance in both tasks [7], [8], [9], [10]. State-of-the-art methods tend to incorporate the attention mechanism into such models [11].

Although the above approaches deal with both intent detection and slot filling problems effectively, training a practical, application-ready model generally requires a large and diversified dataset, which needs to cover most potential user queries. Unfortunately, building such a dataset is often costly and time-consuming, and requires substantial manual effort. The dataset size is dependent on the domain and its scope. Thus, it may be impossible to prepare data manually without the assistance of semi-automatic or automatic methods. Therefore, several tools have been developed to facilitate the creation of datasets for NLU. The typical approach involves describing the problem domain in formal language, typically resembling a context-free grammar. The dataset is then randomly generated based on that grammar. The tools that follow this approach include Chatito[1], Nalgene[2], and SippyCup[3].

Moreover, the architectures that incorporate pre-trained word embedding aid in reducing the need for data, as they perform improved generalisation owing to semantic information being included in the word vectors. The majority of recent methods have relied on word embedding [11], [10], [12], [13]. Kim et al. [10] explored the concept of tuning an embedding layer in order to improve their model performance further. They used external lexicons to enrich a pre-trained word vector model with additional semantic information, which allowed them to improve the accuracy of intent detection. This concept originated from previous works on word embedding [14], [5]. In these studies, the enrichment involved fine-tuning word vector pairs using stochastic gradient descent, by minimising the three max-margin objective functions defined for those pairs. Separate functions have been defined for groups of synonyms, antonyms and related words, obtained from the WordNet[4], ParaPhraseDataBase[5], and Macmillan lexicons[6].

In this study, we propose a novel method for improving model performance by expanding the training set. The proposed method is evaluated using an architecture that covers a long short-term memory (LSTM) layer, an attention mechanism, and CRF. More specifically, the primary objectives of this study are

[1] https://github.com/rodrigopivi/Chatito
[2] https://github.com/spro/nalgene
[3] https://github.com/wcmac/sippycup
[4] https://wordnet.princeton.edu
[5] http://paraphrase.org
[6] https://www.macmillandictionary.com/dictionary/british/lexicon

978-1-7281-4569-3/19/$31.00 ©2019 IEEE

as follows:

- to propose a novel method of expanding a training set dedicated to the case of intent detection and slot filling; and
- to evaluate the proposed method experimentally by using a compound neural framework for supervised classification problems of intent detection and slot filling.

The novelty of our approach relies on expanding a training set based on the random mutation of training samples, following a set of heuristic rules that utilise external lexicons and the training data itself. To the best of our knowledge, this approach has never been attempted in any other studies in the context of intent detection and slot filing. The proposed method is suited to intent detection and slot filling tasks, and does not impose any specific model architecture. The aim is to create a solution for situations in which we have a small set of previously annotated data and wish to avoid expanding them manually. This approach may be preferable for systems in which we are able to collect real queries from users. In such cases, it may be easier to create a small annotated dataset than to prepare a formal grammar for the system, as the former can be implemented in a semi-automatic manner.

The remainder of this paper is structured as follows. Section II defines the problem. Section III describes the architecture of the proposed system for intent detection and slot filling. Section IV presents the proposed method of dataset expansion. Section V discusses experimental studies on our system and compares its performance with that produced by other approaches. Finally, conclusions are provided in section VI.

Please note that we use the following symbol conventions in this paper: x is a variable or constant or function; $\boldsymbol{x}$ is a vector; and $\boldsymbol{X}$ is a matrix or set.

II. Problem formulation

In this section, we define the tasks of intent detection and slot filling.

A sentence can be represented as a sequence of words or subsequences:

$$\boldsymbol{x} = [x_1, ..., x_i, ..., x_I]^T . \tag{1}$$

The problem of intent detection involves mapping this sequence to one or more labels representing the user's general intention. Therefore, we wish to learn a function

$$f : \mathbf{x} \rightarrow \mathbf{y}, \tag{2}$$

where $\mathbf{y}$ may either be a single label or a vector of labels

$$\boldsymbol{y} = [y_1, ..., y_l, ..., y_L]^T , \tag{3}$$

depending on whether it is legitimate for our domain to treat the problem as a multi-label classification task.

Moreover, we define the function

$$g : \mathbf{x} \rightarrow \mathbf{y} \tag{4}$$

for the slot filling task. The function maps each element x_i of an input sequence $\mathbf{x}$ to its label y_l. In general, a beginning-inside-outside (BIO) tagging format is used for slot filling, which allows for assigning a specific label to a subsequence spanning more than one word. In this approach, each word may have a beginning tag (indicating the start of a subsequence of words with the same label), an inside tag (indicating a continuation of a labelled sequence) or an outside tag (assigned to all unlabelled irrelevant words). Thus, two separate sets of possible classes exist, namely (i) $\mathbf{Y}_{IN}$ for the intent detection task

$$\boldsymbol{Y}_{IN} = \{y_1, ..., y_n, ..., y_N\} \tag{5}$$

and (ii) $\mathbf{Y_{SL}}$ for the slot filling task

$$\boldsymbol{Y}_{SL} = \{y_1, ..., y_m, ..., y_M, O\}, \tag{6}$$

where O is the outside tag.

For the purpose of this study, we define

$$\boldsymbol{S}_{SL} = \{(x_i, y_k) : \ k = 1, ..., K\} \tag{7}$$

to be a set of all K labelled sub-sequences originating from the input sequence $\mathbf{x}$ and

$$\boldsymbol{S}_O = \{x_i : \ g(x_i) = O\} \tag{8}$$

to be a set of all outside words in the input sequence. An example of a sentence with its intent and slot labels is illustrated in Figure 1.

III. Framework for intent detection and slot filling

In this section, we present the overall concept of a framework for intent detection and slot filling. In particular, we focus on the framework architecture and processes as well as training issues.

A. Architecture and data flow

An overview of the general architecture of the framework for intent detection and slot filling is provided in Figure 2. We use a neural network model with two separate output layers, in which one layer is used to predict the intent class, while the other is used to predict the slot labels. The application of neural networks was inspired by the literature review provided in section I. We found that the modern networks outperform other approaches utilised for these tasks.

A sequence $\boldsymbol{x}$ of input words is transformed into numeric vectors using an embedding layer. This layer is trainable, as tuning the weights during the training phase is consistently superior to using a fixed layer. The main component of the proposed architecture is the bidirectional LSTM layer [15]. It reads the input sequence $\boldsymbol{x}$ twice, that is, forward and backward. The forward pass reads the original sequence, while the backward pass reads the sequence in the reverse order. At each time-step t, the vectors of the corresponding forward $\boldsymbol{fh}_t$ and backward hidden $\boldsymbol{bh}_t$ states are generated. The final hidden states $\boldsymbol{h}_t$ at time-step t are a concatenation (indicated by a $\oplus$ symbol) of these states, as follows:

$$\boldsymbol{h}_t = \boldsymbol{fh}_t \oplus \boldsymbol{bh}_t \tag{9}$$

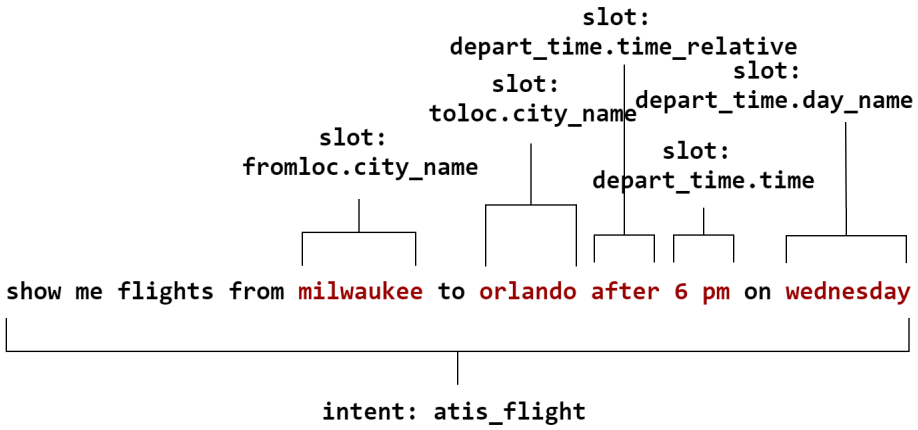

Fig. 1. Example of sentence with intent and slot labels.

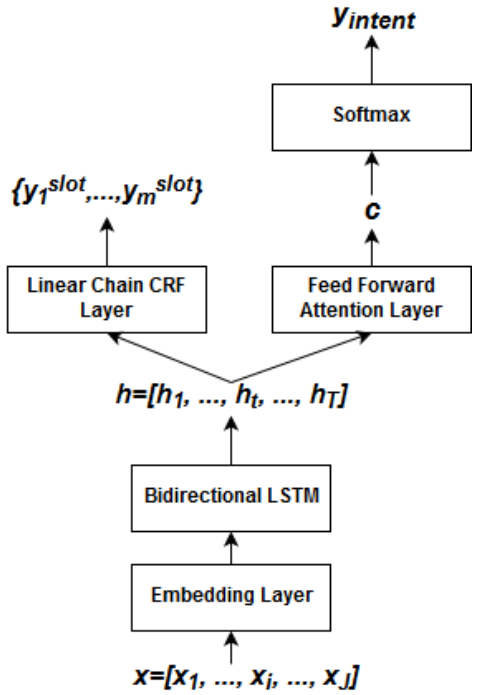

Fig. 2. Architecture overview of framework for intent detection and slot filling.

Since the bidirectional LSTM analyses input sentences from the past to the future and vice visa, it understands their context better than a unidirectional recurrent network, which improves the whole framework performance [15], [16]

Thereafter, the sequence is passed to two layers, responsible for generating slot and intent outputs, respectively. For the purpose of intent detection, we use a feed-forward attention layer [17], which is a type of attention mechanism that computes a fixed-length context vector $\boldsymbol{c}$ as a weighted average of its input vectors for all time-steps $t = 1, 2, ..., T$ as follows:

$$\boldsymbol{c} = \sum_{t=1}^{T} a_t \boldsymbol{h}_t, \tag{10}$$

where a_t is the value of a learnable function computed by another hidden feed-forward layer, and is calculated as follows:

$$a_t(\boldsymbol{h}_t) = \frac{exp(e_t(\boldsymbol{h}_t))}{\sum_{t=1}^{T} exp(e_t(\boldsymbol{h}_t))} \tag{11}$$

$$e_t(\boldsymbol{h}_t) = tanh(\boldsymbol{W}\boldsymbol{h}_t + \boldsymbol{b}_t), \tag{12}$$

where $\boldsymbol{W}$ is a matrix of weights and $\boldsymbol{b}_t$ is a vector of biases. Then, the context vector $\boldsymbol{c}$ is used as input to the output layer with a softmax activation function. The function returns a vector of probabilities for each intent class. A class y_n to which the highest probability p_n is assigned is the intent of sentence $\boldsymbol{x}$:

$$y_{intent} = \{y_n : p_n = max,\ n = 1, ..., N\}. \tag{13}$$

For the slot filling component, we use a linear chain CRF layer, which, for each time-step t, computes the vector of marginal probabilities $P_{m,i,t}(y_{m,t}|\ x_{i,t})$, that is, the probabilities of assigning the label y_m to the word x_i. Thereafter, the layer is optimised with categorical cross-entropy loss to maximise the composition likelihood (product of marginal probabilities). As the model is trained for joint intent detection and slot filling, the errors from all output layers are used to update the model weights. Both layers use a categorical cross-entropy loss function, but for the slot layer, the error is divided by the number of words in the sequence. Therefore, the errors are equally weighted and the model is optimised for both tasks, with no preference for slot filling.

B. Training algorithm and selected parameters

For training, we utilise the gradient descent with ADAM optimisation algorithm [18], as the initial experiments affirmed that the method results in faster convergence for this problem

than the stochastic gradient descent (SGD). During training, we use dropout as a regularisation method. The dropout level was experimentally set to 0.5 for the word embedding layer, and 0.1 for the LSTM and feed-forward attention layers. Higher values often led to unstable convergence of the method, while lower values caused overfitting. Moreover, we set the dimensionality of the hidden states to 100. Thus, the concatenated layer output is a sequence of 200-dimensional vectors. These parameter values produced optimal results in the initial experiments. The model is trained for 50 epochs, as the initial experiments indicated no further improvement for larger values.

The model performance is indicated as the intent accuracy and slot F1 scores, as well as the standard deviations and confidence intervals for those measures (with *p=0.95*). These indicators are calculated by using the 10-fold procedure. As the ATIS dataset uses the BIO format, we follow the original method from the *conlleval* script (CoNLL-2000 shared task) to measure the model F1 score.

IV. Dataset Expansion

We have mentioned that Kim et al. [10] used external dictionaries of synonyms to inject additional semantic information into the model by fine-tuning the word embedding layer. In this study, we test the hypothesis that it is also possible to inject semantic information directly into a training dataset by generating new training samples from existing samples. A new sample is created by applying a random mutation of the existing sample, using some predefined rules. Possible mutations include reordering of the original sequence or replacement of one or more words. In the case of replacement, the rules define a set of possible valid replacements for each considered word sequence, and the new sequence is selected randomly with uniform probability for every set element. One of these rules is based on an external lexicon of synonyms; the others are based solely on the training data. More specifically, we propose three heuristic rules for generating new training samples, as follows.

Rule 1. Random replacement of a labelled slot: We define $\boldsymbol{S}_{seq}$ to be a set of all unique word sub-sequences x_l occurring in the training data and having a label y_l. Given a set $\boldsymbol{S}_{SL}$ of all labelled sub-sequences x_i of an input sequence $\boldsymbol{x}$, we replace each sub-sequence x_i with a randomly selected sequence x_l, with a probability of p_1 from the $\boldsymbol{S}_{seq}$, as follows:

$$\forall_{(x_i,\, y_l)\in \boldsymbol{S}_{SL}}\; x_i \longrightarrow x_i^{'} |\, p_1,\; x_i^{'} = x_l \in \boldsymbol{S}_{seq} \tag{14}$$

In other words, the rule replaces one labelled slot with another random slot, assuming that both slots contain the same label. An example of applying the rule to a sentence would be: "list flights from **philadelphia** to **dallas** on **friday**" $\rightarrow$ "list flights from **los angeles** to **detroit** on **monday**."

Rule 2. Random replacement of an outside word: We define $\boldsymbol{S}_{syn}$ to be the set of all words x_i' that are synonymous with a word x_i; and are reminded that a set $\boldsymbol{S}_O$ covers all outside words of an input sequence $\boldsymbol{x}$. We replace each outside word $x_i \in \boldsymbol{S}_O$ with a randomly selected synonym x_i' selected from $\boldsymbol{S}_{syn}$, as follows:

$$\forall_{x_i\in \boldsymbol{S}_O}\; x_i \longrightarrow x_i' |\, p_2,\; x_i' \in \boldsymbol{S}_{syn} \tag{15}$$

The replacement occurs with probability p_2 and only if $\boldsymbol{S}_{syn} \neq \oslash$.

That is, the rule replaces each outside word with another randomly selected synonym word. An example of applying the rule to a sentence would be: "show me all the **types** of **aircraft**" $\rightarrow$ "show me all the **kinds** of **airplane**."

Rule 3. Sequence order mutation: This rule can be applied to simple sentences containing one labelled sub-sequence and one sequence of outside words. Specifically, if $\mathbf{y} = \left[y_1^{slot}, ..., y_n^{slot}\right]^T$ and there exists such a label c and such an index k, that $\forall_{i<k}\; y_k = c \;\wedge\; \forall_{j\geq k}\; y_k = O$ or $\forall_{i<k}\; y_k = O \wedge \forall_{j\geq k}\, y_k = c$, we change the sequence order by switching the sequences of outside and labelled words. This rule is applied with a probability p_3.

$$\mathbf{x} = [x_1,\, ...,\, x_k,\, ...,\, x_n]^T \longrightarrow \mathbf{x}' = [x_k,\, ...,\, x_n,\, x_1,\, ...,\, x_{k-1}]^T \tag{16}$$

An example of applying the rule to a sentence would be: "list airports **new york**" $\rightarrow$ "**new york** list airports".

We must note that the higher the probabilities (p_1, p_2, p_3) are, the more likely rule 1, 2 or 3 will be applied to the particular sentence $\boldsymbol{x}$. This means that if they are close to one, almost all labelled slots or outside words will be replaced or subsequences mutated. On the contrary, if they are close to zero, a duplicate of the sentence x may be produced, and the rule must be applied again. Overall, high probabilities reduce the number of repetitions required to avoid duplications of sentences and produce more diverse sentences in comparison to their originals. Therefore, we can conclude that the values of probabilities have little impact on the quality of the resulting dataset.

V. Experimental Studies

In subsection V-A, we demonstrate the performance of the proposed methods by means of examples and provide details on the data pre-processing steps and training procedure. Subsection V-B covers experiments on expanding a data set.

A. Multi-label vs. single-label approach

a) Dataset: One of the most commonly used datasets for evaluating intent detection and slot filling methods is that created for the Airline Travel Information System (ATIS) project [19]. The dataset consists of simple queries relating to flight information and air travel. It includes 4,978 training samples and 893 test samples, with 17 unique intent labels and 79 slot labels.

In the ATIS dataset, certain samples may have more than a single intent; thus, it represents a multi-label classification problem. However, several studies have simplified this problem to a single-label classification by selecting only the first label of the multi-labelled samples. As the selected approach affects the classification results, we conducted experiments on the

proposed model twice. Firstly, we applied the single-label approach, in which the output of an intent layer is one of the 17 intent labels from the training set. Secondly, we utilised the multi-label approach, in which we extended the set of labels to include pairs of labels that were present in the training set, by treating each pair as a separate class. This increased the number of intent labels from 17 to 22.

b) Pre-processing: The weights for the word embeddings are initialised with the pre-trained Global Vectors for Word Representation (GloVe) model, which was trained on Wikipedia 2014 and Gigaword 5 data[7], and covers 200 dimensions. The GloVe model suits our task best because, as a count-word based method, it captures the global statistic of a corpus, which may help the framework to infer the intention of sentences better than local, prediction-based methods [20]. Before the data could be used by the model, we had to apply several simple pre-processing steps to the training and test sets. Firstly, we replaced all of the digit sequences with <numberN> tokens, where N represents the number of digits originally present in this sequence (for example, "1920" becomes "<number4>"). Moreover, all unknown words not present in the pre-trained GloVe model were replaced by the <unk> token.

c) Results of a multi-label approach: This subsection provides the results for the multi-label approach and a comparison of these results with those of Kim et al. [10], who followed a similar methodology. Their model achieved an intent accuracy of 97.31% using external dictionaries, or 95.63% without any additional data. The average accuracy of our model was 97.53% with no external data. Both models utilised a pre-trained GloVe word embedding layer, which was proposed by Pennington et al. [20]. The proposed model achieved superior performance when the embedding layer was updated during the training, which is consistent with the results from Kim et al. [10] (see Tables I and II).

TABLE I

RESULTS OF JOINT MULTI-LABEL INTENT DETECTION AND SLOT FILLING (MODEL TRAINED FOR 50 EPOCHS, WITH RESULTS AVERAGED OVER 10 EXECUTIONS).

Measure	Max	Min	Avg	Standard deviation	Confidence interval (p=0.95)
Intent accuracy	97.98	97.20	97.53	± 0.21	97.40 - 97.66
Slot F1 score	95.16	92.45	94.02	± 0.87	93.48 – 94.56

TABLE II

RESULTS OF JOINT MULTI-LABEL INTENT DETECTION AND SLOT FILLING COMPARED TO RESULTS PRODUCED BY PREVIOUS STUDIES.

Model	Intent accuracy	Slot F1 score
This model	**97.53**	94.02
Kim et al. [10]	97.31	Not included

d) Results of a single-label approach: Now, we compare the results of the single-label classification produced by our model with those of other single-label models. Although the authors of certain studies did not explicitly state the approach they used, they are also included. Moreover, we included four slot filling models without intent outputs. In the task of intent detection, the network achieved state-of-the-art accuracy at 98.65%. For the slot filling task, several models (such as the attention-based models proposed by Liu and Lane [11]) achieved superior results (see Tables III and IV).

TABLE III

RESULTS OF JOINT SINGLE-LABEL INTENT DETECTION AND SLOT FILLING (MODEL TRAINED FOR 50 EPOCHS, WITH RESULTS AVERAGED OVER 10 RUNS).

Measure	Max	Min	Avg	Standard deviation	Confidence interval (p=0.95)
Intent accuracy	98.99	98.43	98.65	± 0.16	98.55 – 98.75
Slot F1 score	95.10	93.38	94.64	± 0.48	94.34 – 94.94

TABLE IV

RESULTS OF JOINT SINGLE-LABEL INTENT DETECTION AND SLOT FILLING COMPARED TO PREVIOUS WORKS, WHERE SOME OF THE MODELS ARE SLOT FILLING ONLY.

Model	Intent accuracy	Slot F1 score
This model	**98.65**	94.64
Liu and Lane [11] *(Attention encoder-decoder)*	98.43	95.87
Liu and Lane [11] *(Attention BiRNN)*	98.21	**95.98**
Hakkani-TÃijr et al. [9]	Not included	94.70
Guo et al. [8]*(RecNN)*	95.40	93.22
Guo et al. [8]*(RecNN+Viterbi)*	95.40	93.96
Xu and Sarikaya [7]	94.09	95.42
Tur et al [21]	96.98	95.00
Zhu and Yu [12]	-	95.79
Kurata et al. [6]	-	95.66
Mesnil et al. [13]	-	95.06
Liu and Lane [4]	-	94.89

B. Expansion of training set

a) Dataset mutations: For the purpose of this experiment, we constructed a lexicon consisting of 175 synonym pairs of extracted from the BabelNet multilingual dictionary [22]. The experiment was conducted on randomly generated subsets, composed of 10% to 80% of the original training set as well as the entire training set. For each subset, we trained our model four times (with the multi-label approach): first on the original subset, and then on the datasets expanded by 50%, 100%, and 200%. In the case of 100% and 200%, each training sample was mutated once or twice, respectively. For 50%, half of the training samples were randomly selected and mutated. If the mutation resulted in a sample being a duplicate of an already existing sample, it was rejected and the mutation was repeated until a unique result was obtained. Figure 3 illustrates the expansion procedure in the three exemplary cases, i.e. when

[7]https://nlp.stanford.edu/projects/glove

10%, 50%, and 100% of the training set is used as the initial data. The values of the probabilities p_1, p_2 and p_3 were set to 1.0, 0.75, and 0.50. These values were selected experimentally and the procedure was executed 10 times.

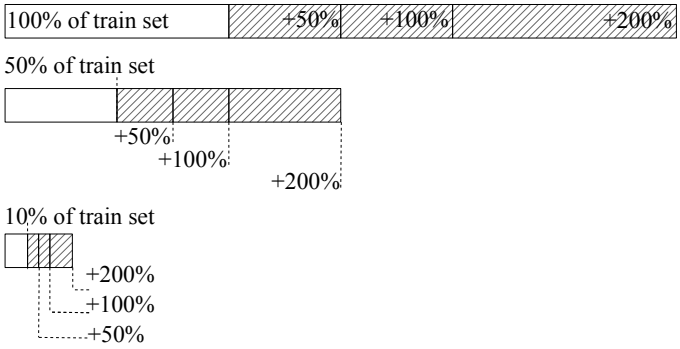

Fig. 3. Illustration of data set expansion starting from 100%, 50%, and 10% of the original training set. The white bars represent the original data; the cross-hatched bars show the scope of expansion. The bars preserve proportions showing the number of samples.

TABLE V
RESULTS OF EXPERIMENTS ON TRAINING SET EXPANSION (MODEL TRAINED FOR 50 EPOCHS; RESULTS AVERAGED OVER 10 RUNS; INTENT - INTENT ACCURACY, SLOT - SLOT F1 SCORE).

Training set –>	10%		20%	
Expansion	Intent	Slot	Intent	Slot
Original	90.17 ±1.04	81.30 ±2.04	93.86 ±0.95	88.04 ±1.27
+50%	89.87 ±0.71	83.00 ±1.75	94.04 ±0.82	90.26 ±0.56
+100%	**90.71** ±0.98	83.63 ±3.33	94.09 ±0.77	91.02 ±2.06
+200%	90.13 ±1.88	**85.31** ±3.34	**94.32** ±1.21	**91.36** ±1.86

Training set –>	40%		60%	
Expansion	Intent	Slot	Intent	Slot
Original	96.53 ±0.50	92.75 ±0.50	97.15 ±0.44	93.90 ±0.68
+50%	96.42 ±0.51	**94.19** ±0.63	97.14 ±0.35	94.16 ±2.10
+100%	**96.62** ±0.76	93.45 ±1.50	**97.28** ±0.20	94.10 ±0.65
+200%	96.45 ±0.51	94.01 ±0.73	96.96 ±0.39	**94.57** ±0.66

Training set –>	80%		100%	
Expansion	Intent	Slot	Intent	Slot
Original	97.53 ±0.31	93.70 ±0.46	**97.53** ±0.19	94.02 ±0.48
+50%	97.44 ±0.28	**94.96** ±0.53	97.42 ±0.20	**94.80** ±0.52
+100%	**97.54** ±0.29	94.38 ±0.58	97.51 ±0.18	94.52 ±0.40
+200%	97.27 ±0.27	94.35 ±0.73	97.26 ±0.34	94.47 ±0.80

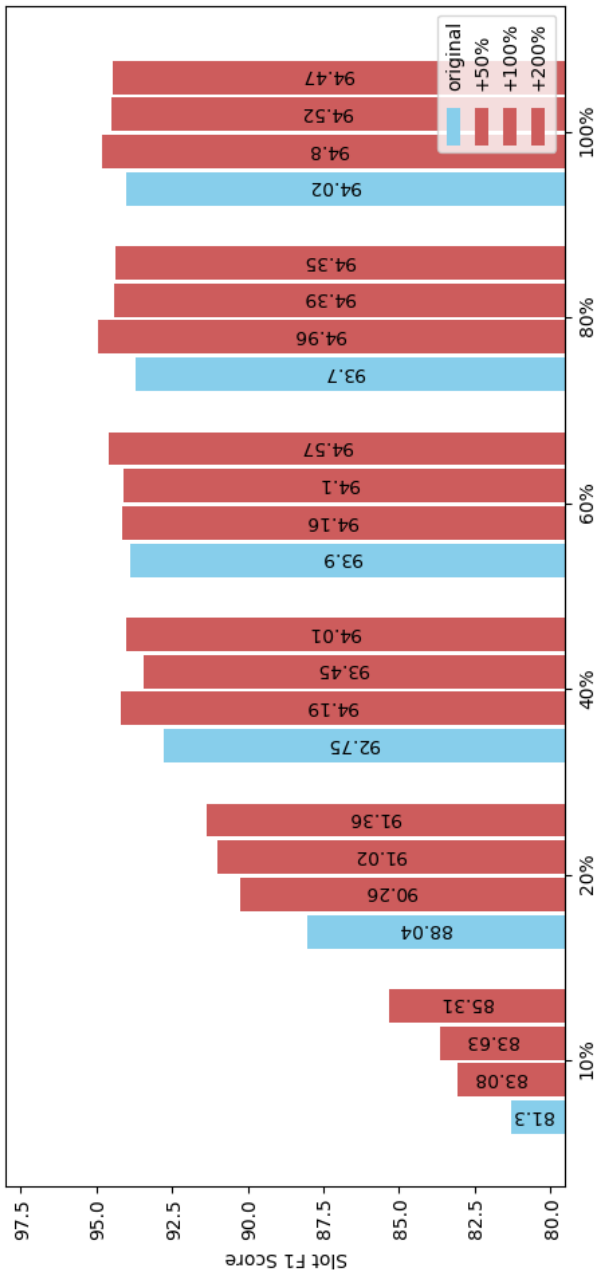

Fig. 4. Slot F1 scores for original and expanded datasets from Table V.

b) Results: Table V displays the average scores for the intent accuracy and slot F1 score. The results indicate that the expansion of the training set had a limited impact on the intent accuracy (see Table V). In each case, the accuracy of the models trained on the expanded datasets was similar to the accuracy of the model trained on the original dataset. However, an improvement was apparent in the slot F1 score (see Table V and Figure 4). The increase is significant for each training subset size, but apparently more significant for smaller datasets. For the subset of 10% of the training data, the score improved from 81.30 to 85.31 (by 4.01%). For the larger subsets, the differences decreased. For the full training set, the maximum improvement was 0.78%. Nevertheless, for every dataset size, the expanded models provided consistently higher scores. This experiment demonstrates that it is possible to inject semantic information not only through the word

embedding layer, but also through training set modification, and the latter method also contributes to improving the model results. Furthermore, this method can be used for a wider variety of models, including those that do not utilise embedding layers.

VI. Conclusions

In this paper, we have developed a compound neural architecture for slot filling and intent detection. It is composed of a long short-term memory layer, an attention mechanism and conditional random fields, which inherits from state-of-the-art approaches and enriches them as well. Moreover, we proposed a novel method for training set expansion that can be used to generate data automatically for the above-mentioned tasks. We have demonstrated that the proposed model outperforms previous approaches to intent detection by comparing the classification accuracy.

Through illustrative experiments, we have shown that expanding the training set by applying a set of randomised heuristic rules can improve the model performance for the slot filling task. As the AITS dataset was utilised in related studies, we validated our approach only on these data to provide comparable results. The experimental results dataset indicate that the improvement is more significant for small datasets, but may be beneficial even for larger sets. This simple method can be used to boost model performance or reduce the amount of data required to train the model without the effort of manually labelling additional examples.

Further research may include ablation studies examining the influence of each heuristic or generic operator on the final performance of an intent detection and slot filling model. Another study may focus on improving a deep neural architecture which can lead to better natural language understanding.

References

[1] J. D. Koppelman, "A statistical approach to language modelling for the atis problem," Cambridge, MA, USA, Tech. Rep., 1995.

[2] M. Jeong and G. G. Lee, "Exploiting non-local features for spoken language understanding," in *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, ser. COLING-ACL '06. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006, pp. 412–419. [Online]. Available: http://dl.acm.org/citation.cfm?id=1273073.1273127

[3] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[4] B. Liu and I. Lane, "Recurrent neural network structured output prediction for spoken language understanding," in *NIPS Workshop on Machine Learning for Spoken Language Understanding and Interactions*, 2015.

[5] N. Mrksic, D. Ó. Séaghdha, B. Thomson, M. Gasic, L. M. Rojas-Barahona, P. Su, D. Vandyke, T. Wen, and S. J. Young, "Counter-fitting word vectors to linguistic constraints," *CoRR*, vol. abs/1603.00892, 2016. [Online]. Available: http://arxiv.org/abs/1603.00892

[6] G. Kurata, B. Xiang, B. Zhou, and M. Yu, "Leveraging sentence-level information with encoder LSTM for natural language understanding," *CoRR*, vol. abs/1601.01530, 2016. [Online]. Available: http://arxiv.org/abs/1601.01530

[7] P. Xu and R. Sarikaya, "Convolutional neural network based triangular crf for joint intent detection and slot filling," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec 2013, pp. 78–83.

[8] D. Guo, G. Tur, W. t. Yih, and G. Zweig, "Joint semantic utterance classification and slot filling with recursive neural networks," in *2014 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2014, pp. 554–559.

[9] D. Hakkani-Tï£¡r, G. Tur, A. Celikyilmaz, Y.-N. V. Chen, J. Gao, L. Deng, and Y.-Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm." ISCA, June 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/multijoint/

[10] J. K. Kim, G. Tur, A. Celikyilmaz, B. Cao, and Y. Y. Wang, "Intent detection using semantically enriched word embeddings," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 414–419.

[11] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *CoRR*, vol. abs/1609.01454, 2016. [Online]. Available: http://arxiv.org/abs/1609.01454

[12] S. Zhu and K. Yu, "Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5675–5679.

[13] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, March 2015.

[14] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. H. Hovy, and N. A. Smith, "Retrofitting word vectors to semantic lexicons," *CoRR*, vol. abs/1411.4166, 2014. [Online]. Available: http://arxiv.org/abs/1411.4166

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Compututation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[16] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[17] C. Raffel and D. P. W. Ellis, "Feed-forward networks with attention can solve some long-term memory problems," *arXiv preprint arXiv:1512.08756*, vol. abs/1512.08756, 2015. [Online]. Available: http://arxiv.org/abs/1512.08756

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[19] P. J. Price, "Evaluation of spoken language systems: The atis domain," in *Proceedings of the Workshop on Speech and Natural Language*, ser. HLT '90. Stroudsburg, PA, USA: Association for Computational Linguistics, 1990, pp. 91–95. [Online]. Available: https://doi.org/10.3115/116580.116612

[20] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[21] G. Tur, D. Hakkani-Tür, L. Heck, and S. Parthasarathy, "Sentence simplification for spoken language understanding," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5628–5631.

[22] R. Navigli and S. P. Ponzetto, "Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network," *Artificial Intelligence*, vol. 193, pp. 217 – 250, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0004370212000793