Neural Computing and Applications (2021) 33:5705–5718
https://doi.org/10.1007/s00521-020-05351-2

ORIGINAL ARTICLE

Semi-supervised deep learning based named entity recognition model to parse education section of resumes

Bodhvi Gaur[1,2] · Gurpreet Singh Saluja[1] · Hamsa Bharathi Sivakumar[1] · Sanjay Singh[1]

Received: 8 February 2020 / Accepted: 7 September 2020 / Published online: 18 September 2020

Abstract
A job seeker's resume contains several sections, including educational qualifications. Educational qualifications capture the knowledge and skills relevant to the job. Machine processing of the education sections of resumes has been a difficult task. In this paper, we attempt to identify educational institutions' names and degrees from a resume's education section. Usually, a significant amount of annotated data is required for neural network-based named entity recognition techniques. A semi-supervised approach is used to overcome the lack of large annotated data. We trained a deep neural network model on an initial (seed) set of resume education sections. This model is used to predict entities of unlabeled education sections and is rectified using a correction module. The education sections containing the rectified entities are augmented to the seed set. The updated seed set is used for retraining, leading to better accuracy than the previously trained model. This way, it can provide a high overall accuracy without the need of large annotated data. Our model has achieved an accuracy of 92.06% on the named entity recognition task.

Keywords Named entity recognition (NER) · Semi-supervised learning · Deep learning models · Natural language processing · Resume information extraction

1 Introduction

Globally, companies receive resumes in large numbers that require screening. Resumes carry semi-structured text, which is difficult to parse. The difficulty arises from differences in structures, styles, formats, order, and types of information that the resumes incorporate. It usually consists of various sections that reflect the candidate's competency. Accurate parsing of these resume sections without manual intervention is a dire need.

A widely used technique for recognizing entities is named entity recognition (NER). NER refers to identifying all the occurrences belonging to a specific type of entity in the text. NER tasks require a large amount of annotated data that could be extremely cumbersome to produce. There exists a need for auto-annotated data, which can provide good accuracy.

One of the most common approaches used for NER is a reference from a list [18, 35]. This approach usually leads to better performance and depends on the entire list and, therefore, defaults. We can also perform NER tasks using various deep learning models.

In NER, the combination of word embedding [23, 24], convolutional neural networks (CNN) [13], bidirectional long-short term memory (Bi-LSTM) [14] and conditional random fields (CRF) is the most preferred combination [27]. In our model, we have used the combination without the CRF layer. CRFs perform better with structured data.

✉ Sanjay Singh
sanjay.singh@manipal.edu

Bodhvi Gaur
bgaur1@jhu.edu

Gurpreet Singh Saluja
gurpreet.singh11@learner.manipal.edu

Hamsa Bharathi Sivakumar
hamsa.bharathi@learner.manipal.edu

1 Department of Information and Communication Technology, Manipal Institute of Technology, MAHE, Manipal 576104, India

2 Present Address: Department of Computer Science, Johns Hopkins University, 3400 North Charles Street, Baltimore, MD 21218, USA

Since resumes have semi-structured data, we decided to skip the CRF layer [5].

1.1 Resume parsing

Much research has been carried out in the field of resume parsing in recent times. Jiang et al. [16] have used statistical and rule-based algorithms for extracting relevant information. However, this approach fails to generalize for resumes in English. Farkas et al. [10] have devised an application where the user uploads his resume from which details are automatically extracted, and an application form is subsequently filled. The user is then allowed to edit the form, if required, and submit it. This method relies on high recall so that even if the information fetched is not precise, the user can edit the automatically extracted information in the form and submit it. A CRF-based resume miner to extract information provides a method for ranking applicants for a given job profile [31]. These methods give low precision and low recall for institute and degree names [10, 31].

A cascaded hybrid model for resume parsing uses a combination of the hidden Markov model and support vector machine to extract information in a hierarchic manner [36]. This method again suffers from low precision and recall for institute and degree names. Pawar, Srivastava, and Palshikar [25] have developed an unsupervised algorithm for automatically creating a gazette. They use a search algorithm for the NER task, which performs better than a naïve approach based on regular expressions.

Jiang et al. [16] have designed a parser, which first partitions the resume with the help of rule-based regular expressions by analyzing the characteristics of a Chinese resume. For necessary information, squeeze and sliding window algorithms have been used to achieve 87% accuracy. For complex information, SVM was trained with 1200 resumes, and on testing with 300 resume test samples, 81% accuracy was obtained. This approach involves more rules than automatic information extraction.

Chuang et al. [8] have leveraged the characteristics of the Chinese resume, wherein it is divided into simple and complex items by an iterative process. Eigenvectors, TF-IDF, and SVM algorithm further identify the multiple items with an average accuracy of 81%.

The task of recognizing the education section of a resume has also been taken up. Ravindranath et al. [29] have taken Gibb's Sampling approach to recognize the education section and other parts such as the work experience of a resume by converting them to a parse-tree. Tikhonova and Gavrishchuk [32] have used NLP-based methods to recognize the education section of a resume. A Jaccard score of 0.806 was achieved for a Russian dataset for resumes.

Sayfullina et al. [30] have presented a novel technique for resume classification into 27 different categories using domain adaptation. An attempt to overcome the paucity of labeled resume data has been made. Three different kinds of resume component datasets have been worked on, namely job descriptions, resume summaries, and children's dream job descriptions, which vary majorly. Two models have been considered, a word embeddings-based fastText method for classification and a CNN model. For each of the categories, mentioned CNN outperforms the fastText method. It opens prospects for future work in improving the results using the CNN model for the low number of labeled resumes available.

1.2 Named entity recognition using deep learning

Recently deep learning-based methods have also been explored for NER. A pre-trained word-embedding is used as an input to a neural network model and character-level features [19, 27]. A comparison of a Bi-LSTM cum CRF model with a transition-based chunking model with shift-reduce parsers is made for NER, concluding that the former gave a better performance [19]. Stacking of recurrent neural network layers for a biomedical sequence was employed for classification purposes [34]. Bi-LSTM, CNN, and CRF layers have been incorporated into the neural network model for unstructured data [26, 27]. Transfer learning is adopted to solve the labeled data scarcity issue, where an artificial neural network (ANN) trained on a large dataset is used to predict another large dataset [20]. For the problem of manual annotation, a semi-supervised approach with bi-lingual corpora is made use of for increasing annotated training data [37].

Yu et al. [36] have developed a cascaded information extraction (IE) framework. A CV is segmented into blocks with labels for different information types in the first pass using an HMM Model. Then in the second go, detailed information, like Name, is extracted from individual blocks instead of searching in the entire resume for it using a hybrid model consisting of HMM and SVM.

Maheshwary and Misra [21] have proposed a Siamese adaptation of CNN to accomplish the task of matching resumes with a particular job opening. The model consists of a pair of identical CNN, which they propose gives them a measure of semantic relatedness of words in the resume in a controllable manner with low computational costs. They test their model against simple models like Bag of Words, and TF-IDF compared to which their model performed better.

Deep learning model approaches have shown a significant performance and accuracy improvement for named entity recognition task. Moreover, it can be generalized over a wide variety of data, unlike the rule-based approaches.

1.3 Named entity recognition for semi-structured data

Chifu et al. [6] have created skills, and web crawled resumes are checked for POS patterns after text preprocessing using the Stanford NLP framework. If words are not present in the skill ontology, new skills are updated for further skill detection using algorithms trained for specific lexical patterns. Wikipedia is the primary source of ontology, and the whole system is highly dependent on the same.

Ghufran et al. [11] leverage the fact that an individual's resume contents are available on Wikipedia for automatic annotation without POS tags. N-grams are constructed from keywords in a resume and then queried to Wikipedia. Returned results are in the form of an interpretation graph, processed for disambiguation and cross-language references. On 153 resumes, education section entities are recognized with an F-score of 85.68%. Dependency on Wikipedia for information is very high, and the dataset of resumes is also limited.

Zhang et al. [8] have proposed a technique for parsing the semi-structured data of the Chinese resumes. The system consists of the following key components, firstly the set of classes used for classification of the entities in the resume, secondly the algorithm used for identification of those entities, and lastly, the system design. The entities are divided into two major subcategories, that is, simple items like name, and date of birth, and miscellaneous items like the learning experience, skills, etc., which exhaustively cover all entities. A total of 5000 resumes is used, and a system comprising of SVM, regular expression, and vector space model-based classifier base has been implemented. The overall accuracy of 87% for necessary information and 81% for complex information has been achieved. Future work opens potential in improving the rough segmentation accuracy in resumes, thereby leading to improved information extraction.

Zhang et al. [38] have proposed an analytic system for the mining and visualizing the semi-structured data in resumes. The semantic information is first extracted after which visualization, in the form of understanding the career progression of an individual, assessing the social relationships, and holding an overall view of the resume is done to represent this collected information. Prospects include incorporating visualization of geographical dimensions.

Darshan [9] has used Perl-based regular expressions to convert semi-structured into an ontological structure. This semantic information is further represented in XML format for information extraction from resumes.

The limited literature on resume parsing has not demonstrated very high accuracy in identifying institutions and degrees [10, 16, 36].

In our work, we focus on accurate identification of academic degrees and institute names in a resume's education section. The proposed method eases the recruiter's search for candidates from specific institutions or academic qualifications. It further helps in the analysis regarding recruitment trends specific to colleges, compensations, and industry exposures provided to the candidates [15].

The main contributions of this paper are summarized as follows:

- We demonstrated the use of a modified semi-supervised technique for parsing institute and degree names. Instead of following the traditional semi-supervised approach, we introduced a correction module to rectify the predictions. We added these corrected predictions back to the original seed set, thereby increasing its size. On retraining, this procedure results in improved accuracy, precision, and recall in comparison with the previously trained model.
- We achieved high performance for recognizing degrees and institutes in a resume without large annotated data.

2 Methodology

In this section, we explain the different modules of the proposed method.

2.1 Preprocessing

The seed set contains 550 resumes, which are split into 50 for testing and 500 for training purposes. The preprocessing included the following steps:

- Conversion of pdf resumes to JSON using PDF2JSON [39]
- Extraction of words from JSON for Part-of-Speech (POS) tagging using the Natural Language Toolkit (NLTK) tagger [3, 4]. POS tags were used as a feature since the identification of proper nouns in the data aids in the identification of institute names.

2.2 Corpus

BILOU is an encoding schema where the last token of multi-token chunks is denoted explicitly by a last (L) tag. The BILOU encoding scheme suggests to learn classifiers that stands for the **B**eginning, the **I**nside and the **L**ast tokens of multi-token chunks as well as **U**nit-length chunks [28]. It means an increase in the number of parameters to be learned by the model. Since we have followed a semi-supervised approach with initially less tagged data, we decided not to burden the model with additional parameters. The BIO (Beginning, Inside, Outside) encoding schema could represent the same data without any additional tag. Hence we choose it to retain accuracy.

We manually annotated the seed set using BIO encoding. The annotations are given in Table 1. From the tagged resumes, we extracted the education sections, which constitutes our raw corpus.

2.3 Data cleaning

When resumes in PDF format are converted to JSON, many inconsistencies can arise. Since the irregularity can arise in many variations, rectifying it to bring them all to a common format is necessary for the model to function properly. Some of those issues are:

- resolving unbalanced parenthesis
- Irregular spacing between words
- Replacing&amp with&, and
- Removing unwanted characters (including non-ASCII characters)

1. **Resolving unbalanced parenthesis**
   Inconsistencies of this kind can be resolved as done in the example below. If the resume contains "Studied in CMS(RDSO)", here the extracted words would be: "Studied", "in", "CMS(", "RDSO)". Ideally, we would want the last two words to be either "CMS", "RDSO" or "CMS", "(RDSO)". In this case, the solution is to check the extracted words for balanced parenthesis and remove the unbalanced ones. The clean function would then return "Studied", "in", "CMS", "RDSO" (Fig. 1).
2. **Irregular spacing between words**
   If the resume contains "Studied in CMS", here the extracted words would be: "Studied", "in", " ", "CMS". The extra space present here between "in" and "CMS" would create one extra empty word which is corrected by ignoring empty words in the document (Fig. 2).
3. **Replacing& with&**
   When the resumes in their PDF form are converted to JSON, the '&' symbol is sometimes returned as '&' and sometimes as '&'; therefore, before moving ahead, it is essential to resolve any inconsistencies of this kind. Consider the following example; if we have "Studied 10th & 12th from CMS" in our resume, the extracted words would be: "Studied", "10th", "&", "12th", "from", "CMS". Here the "&" is replaced with "&" for maintaining consistency among all such cases (Fig. 3).
4. **Removing unwanted characters (including non-ASCII characters)**
   There may be some extra unwanted characters present in the words that are extracted. These extra characters must be removed to ensure a common format. Take the following example; if we had "B.Tech: Computer Science from CMS" in our resume, then the extracted words would be: "B.Tech:", "Computer", "Science", "from", "CMS". Here the unwanted character ':' at the end of the degree 'B. Tech' is removed to give us "B.Tech". Similarly, extra unwanted characters occurring at the beginning of the extracted words are also removed (Fig. 4).

Table 1 Tags and meaning

Tag	Meaning
U-INST	Single word institute name
U-DEG	Single word degree name
B-INST	Start of an institute name
I-INST	Continuation of the institute name
B-DEG	Start of a degree name
I-DEG	Continuation of the degree name
O	Others

2.4 Model

In this subsection, we describe the various components of the proposed model (see Fig. 5).

2.4.1 Classification model

The classification model consists (see Fig. 6) of the following layers implemented using Keras [7]:

- **Word Embedding Layer:** Words used in resumes such as the institute names or the degree names may not be present in a pre-built word embedding. Therefore, we created a new set of word embedding built on our corpus with Keras's help. We produced two-word embedding layers, one for classification entities, and another for their respective POS tags. We then concatenated these two to form the base model. We added a Dropout layer to prevent over-fitting of the data with a probability of 0.1 as experimentally that gave us the best result. We have used 10% of the dataset as the development set.

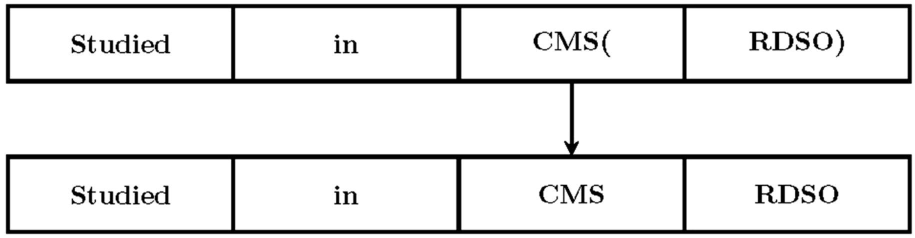

Fig. 1 Resolving unbalanced parenthesis

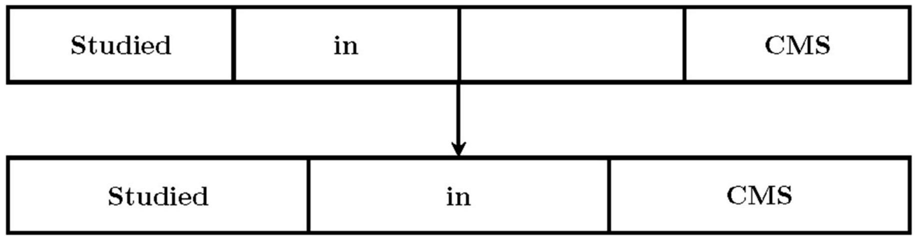

Fig. 2 Irregular spacing between words

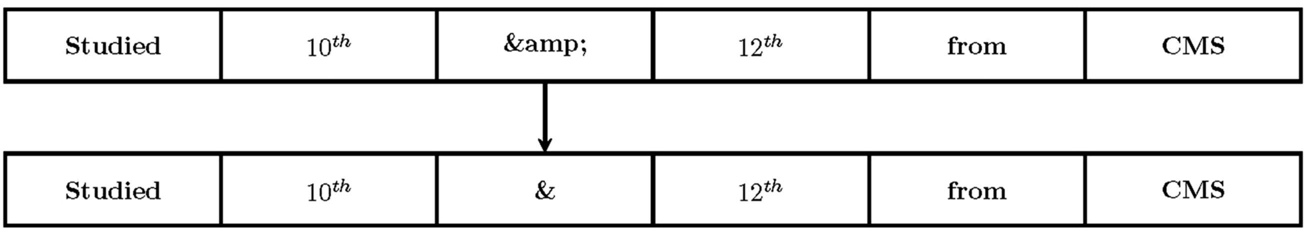

Fig. 3 Replacing & with &

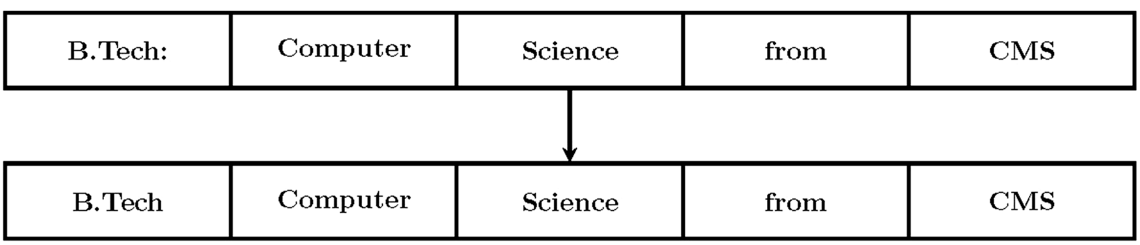

Fig. 4 Removing unwanted characters (including non-ASCII characters)

- **CNN Layer:** A 1-D convolution (since the text is linear data) neural network layer for extracting character level features is further concatenated to the above base.
- **Bi-LSTM Layer:** We appended a Bi-LSTM layer comprising of 100 hidden neurons to the model.

A considerable batch size reduces the generalization ability of a deep learning model [17]. We, therefore, trained the model using a small batch size of five for 20 epochs. We observed that the model converges well in 20 epochs, as shown in Fig. 7.

In Fig. 7 we have used cross-entropy loss as the loss function. Cross-entropy loss measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverge from the actual label. A perfect model would have a cross-entropy loss of 0. The cross-entropy loss function used is given by

$$loss = -\sum_{i}^{K} y_i \log(y_i'),$$

where y_i is the ground truth and y_i' is the predicted score for each class $i \in K$.

2.4.2 Correction module (Processing predictions made on the unlabeled dataset)

The unlabeled data (360 files) undergoes the same cleaning as specified above. We split the unlabeled data into 12 sets of 30 records each. We rectified the classification model predictions made on each set using a correction module. This module's overall task is to ensure that the tags

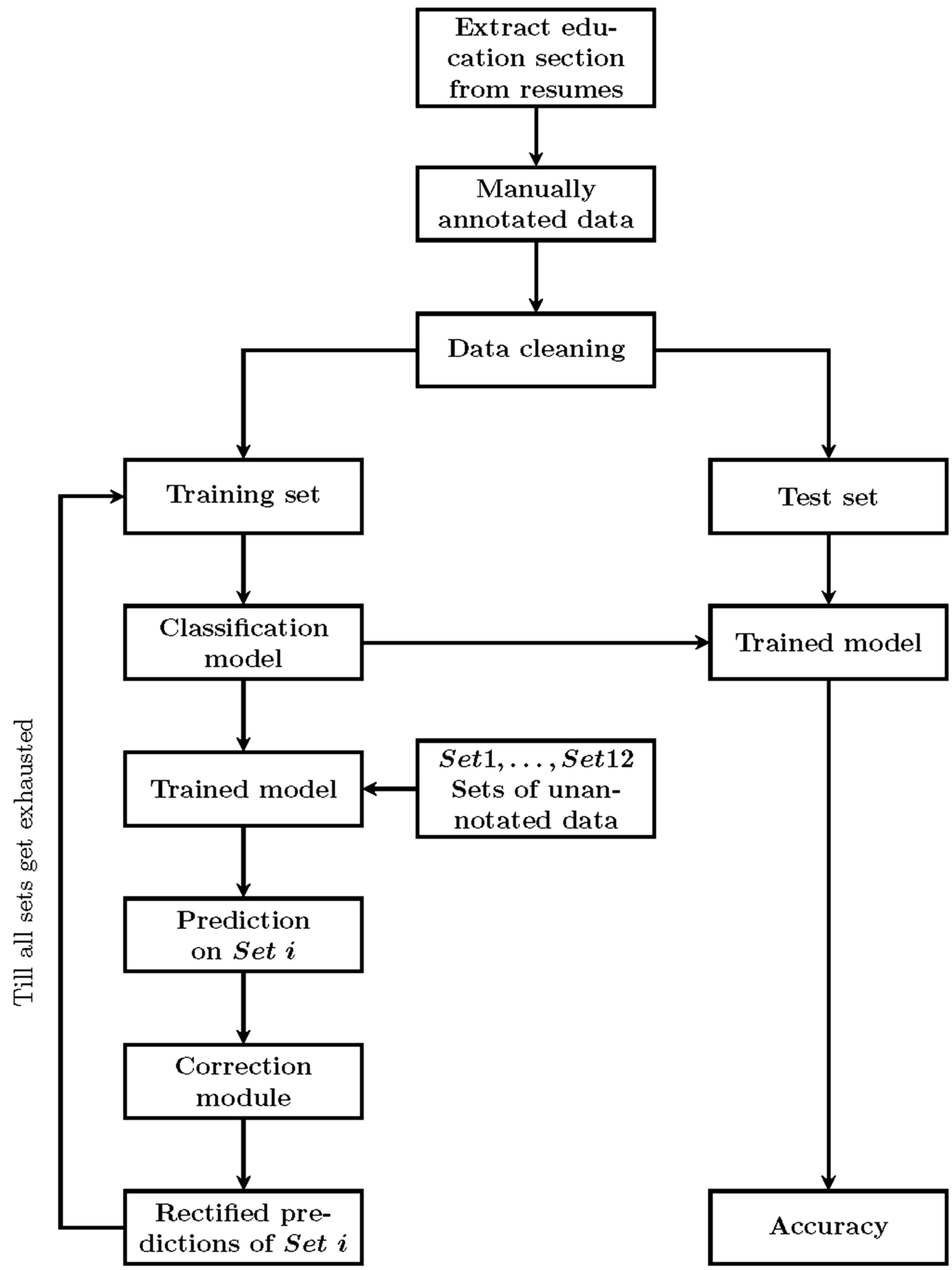

Fig. 5 Proposed model

predicted wrongly could be corrected with the help of a list L1 (Institute names) and L2 (Degree names). The list L1 consists of names of 47,000 colleges and 35,000 schools from India and 23,000 schools and 5300 colleges from USA and L2 consists of 590 degrees collected from various Internet sources.

We used two pointers to extract the initially predicted entity, to check for correctness, as depicted in Fig. 8. A tag that starts with a 'B' or an 'I' is marked with a start pointer. The subsequent end of the tag is marked with the end pointer. The model extracted entities present between these pointers for further comparisons.

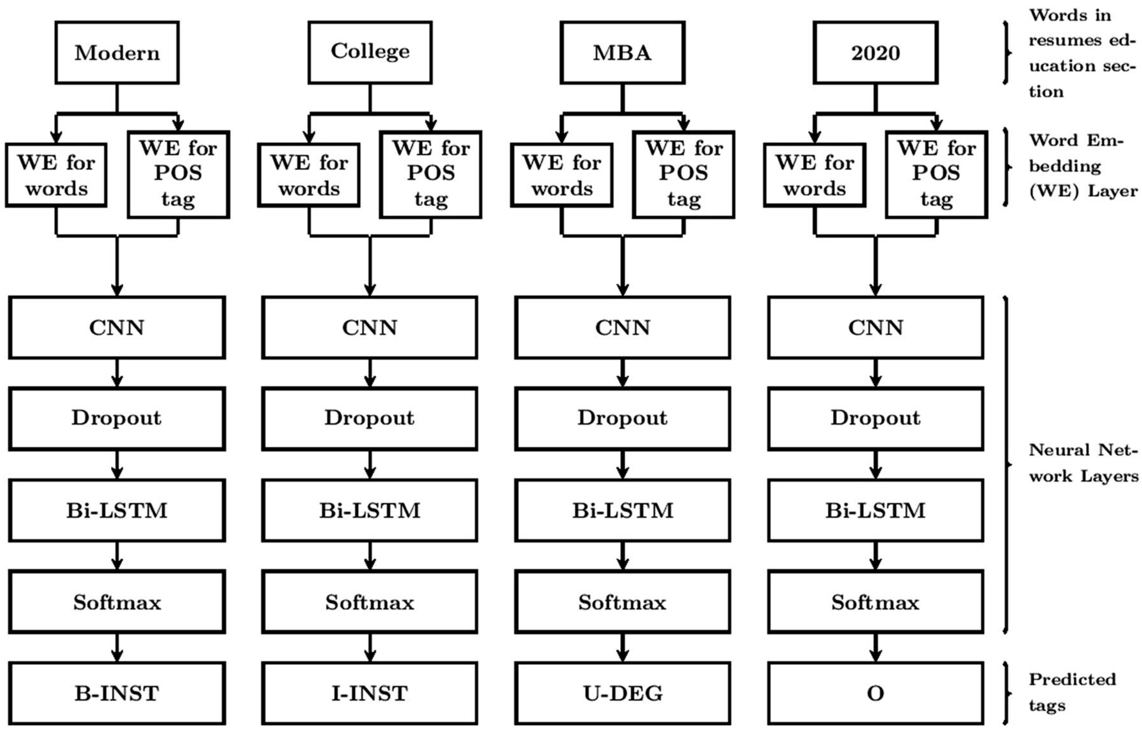

Fig. 6 Deep learning architecture

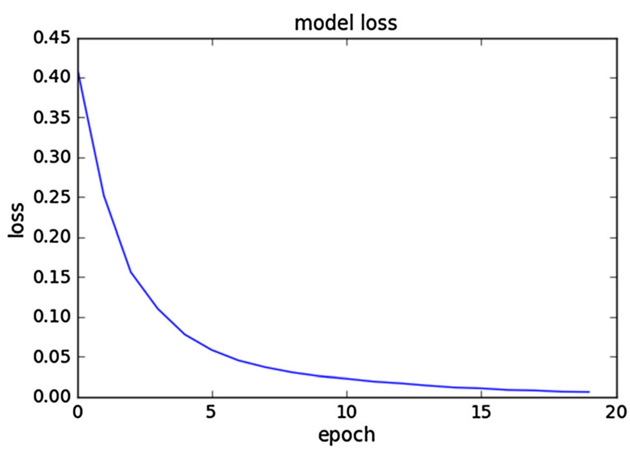

Fig. 7 Loss vs. epoch graph

The two lists generated by us are a comprehensive collection of the institute and degree names in the USA and India, so the trivial approach would be to opt for direct string matching of the extracted entities with the list. However, we aim to create a model for recognizing institute and degree names for any resume in the English language that may not belong to the USA or India. If we use the direct string-matching approach, we may encounter a name that may not be present in our list, since it relies solely on the name in the list for its recognition.

Our model incorporates a neural network component, which does not depend on the list to recognize entities in the resume data and instead works on identifying patterns in the data. Thus, it can successfully recognize new names with similar patterns even if they are not present in the list.

We created a dictionary for mapping short-forms present in institution and degree names to a standard uniform format. For example, we mapped short-forms “engg.” and “eng.” to the word “engineering,” “tech.” to “technology” and “edu.” to “education.” The mapping for short-form has ensured that, if the model found short-forms in either the education section of a resume or the list, it compared it uniformly.

The original resume’s data is not transformed: these mappings are used to aid the correction module in comparing the original name found in the list with the extracted entity. It is done to reduce the error in tag-prediction. For example, while comparing the entities of the institute ABC Institute of Tech. to the entry in the list-ABC Institute of Technology, we map the word Tech. to Technology to

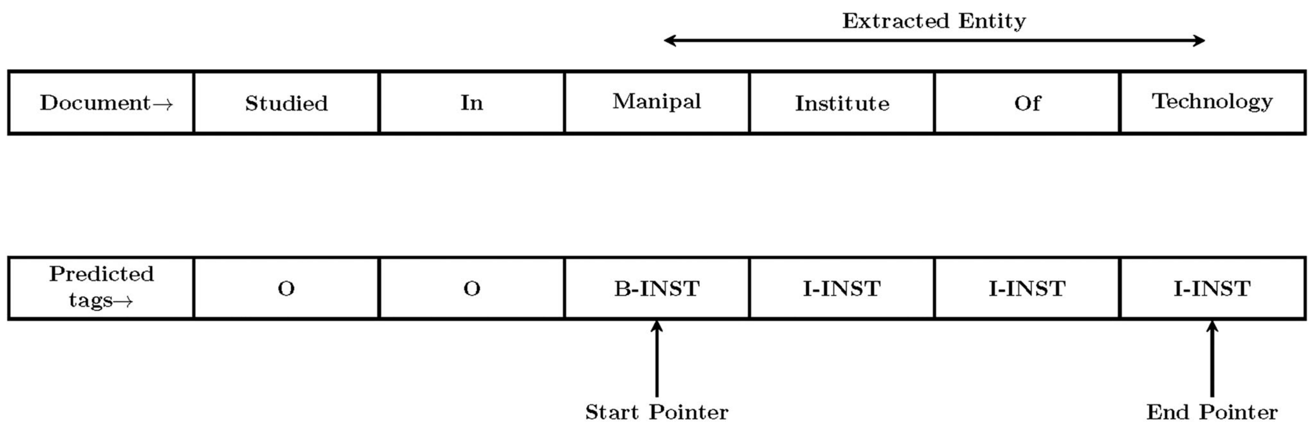

Fig. 8 Extracted entity

resolve the ambiguity. It ensures that without making any change to the data (i.e., resumes), our model can handle real-world data with such ambiguities. It means that our model can identify the institute or degree names even if they are present in a different format.

Since the extracted entity could be incomplete, we employed Fuzzywuzzy [12], a search system to select the best match for institute or degree name in L. Fuzzywuzzy is an open-source tool for searching in a fuzzy manner. Python's Fuzzywuzzy library is used for measuring the similarity between two strings. To obtain the similarity ratio between two strings, it uses Levenshtein distance [2]. We fed Fuzzywuzzy with institute and degree names and the extracted entity to fetch the top five closest matches. Words - "and", "of", "in", and "&" are ignored during the comparison. We noticed that many times people interchange the 'and' and '&' and sometimes they forget the 'in' or 'of' and hence ignored these.

A comparison with the list is made in two ways depending upon whether the name consists of punctuations. If the name does not contain punctuations, the model makes word-level comparisons in the range of $\pm$ five words, from both the pointers and the fetched results. Once the model finds an exact match, we rectify the predicted entity's tags using B-INST/B-DEG for the first word and I-INST/I-DEG for the rest of the entity's words.

For names with punctuations, the model does character level matching making the name fetched from fuzzy-wuzzy and the extracted entity uniform (standard format). The uniform format overcomes variations caused due to the placement of punctuations in the names like "M.H. Institute of Technology" or "M H Institute of Technology" or "M. H. Institute of Technology."

If no exact match, for the results fetched from $L_1(L_2)$, is found, the following procedure is followed:

1. The extracted entity is expanded to accommodate its immediate neighboring word on the left and the right, that is, the start pointer = start pointer-1 and end pointer = end pointer + 1.
2. This entity is then sent to fuzzy-wuzzy to fetch a new set of top five matches w.r.t. this entity.
3. The process mentioned above is repeated to check if an exact match can be found.
4. If it is found, then correct the corresponding predictions in the document.
5. Else, the other list $L_2(L_1)$ is checked.
6. If not even found in the other list, then make the whole predicted entity O (others).

Entities that are already corrected are not rechecked in this entire iterative process for each document. For degree name correction, the same procedure, as mentioned above, is repeated. The details of the correction module are given in Algorithm 1.

Algorithm 1 Correction Module

```
function CORRECTION(D, sP, eP, P, Ins, Deg, S)
    ▷ D:Resume's education section containing multiple tokens
    ▷ sP:Start position of the originally predicted entity in the document marked by a pointer
    ▷ eP:End position of the originally predicted entity in the document marked by a pointer
    ▷ P = {p_1, p_2, ..., p_n}:Predictions corresponding to the entity extracted
    ▷ Ins:List of institutes
    ▷ Deg:List of degrees
    ▷ S:Mapping of short forms to a common form
    ExactMatch ← ∅
    Counter ← ∅
    CounterList ← ∅
    E = {e_1, e_2, ..., e_n}    ▷ Entity extracted from the document in the range sP to eP
    for e ∈ E do
        if e contains a short form or abbreviation then
            Replace e with its corresponding entry in S
        end if
    end for
    if P corresponds to INST then
        L_1 = Ins
        L_2 = Deg
    else if P corresponds to Deg then
        L_1 = Deg
        L_2 = Ins
    end if
    R = FuzzyWuzzy(E, L_1)  ▷ Return a list of top 5 matches for the extracted entity
    Repeat:
    ExactMatch = CheckExactMatch(R, E, sP, eP)                    ▷ Algorithm 2
    if ExactMatch == 1 then
        correct the predictions for the tokens corresponding to the correctly recognized entity
        ▷ First word with B- tag and the rest with I- tag or if just a single word then use U-tag
        Change the tag to O if none of B/I/U applies
    end if
    if ExactMatch == 0 then
        if CounterList == 1 then
            No match found for the extracted entity in both the lists
            Make all the tags corresponding to the extracted entity O
        else if CounterList == 0 then
            ToRepeat = CheckVariations(D, sP, eP, L_1, L_2, Counter)
            ▷ Algorithm 3
            if ToRepeat == 1 then
                goto Repeat
            end if
        end if
    end if
    return E^I, P^I              ▷ Verified entities and corresponding corrected predictions
end function
```

Algorithm 2 Checking Exact Match for the Extracted Entity

```
function CheckExactMatch(R, E, sP, eP)
    ExactMatch = 0
    for r ∈ R do
        if r contains punctuations then
            Compare r (without punctuations) and E (without punctuations) at character level
            if an exact match is found then
                ExactMatch ← 1
                break
            end if
        else if r doesn't contain punctuations then
            Compare r with E (of size n) obtained over the range sP − 5 to eP + 5
            if an exact match is found then
                ExactMatch ← 1
                break
            end if
        end if
    end for
    return ExactMatch
end function
```

Algorithm 3 Checking for Possible Variations in Extracted Entity

```
function CheckVariations(D, sP, eP, L1, L2, Counter)
    if Counter == 1 then                ▷ Exact match is not found for all feasible
        ▷ variation in length over the extracted entity E with the list L1
        CounterList ← 1
        R = fuzzywuzzy(E, L2)           ▷ Go to the other list to look for a match for the extracted entity
        ToRepeat = 1
    else if Counter == 0 then           ▷ checking for matches by increasing the size of the originally extracted entity by one word on both sides
        Counter ← 1
        E2=entity obtained from range sP − 1 to eP + 1 in the document D of size n + 2
        R = fuzzywuzzy(E2, L1)
        ToRepeat = 1
    end if
    return ToRepeat
end function
```

Table 2 Institute name cases covered by correction module

Case 1							
Document	Studied	In	Manipal	Institute	Of	Technology
Predicted tags	O	O	B-INST	I-INST	I-INST	O
Corrected tags	O	O	B-INST	I-INST	I-INST	I-INST
Case 2						
Document	Studied	In	Manipal	Institute	Of	Technology
Predicted tags	O	B-INST	I-INST	I-INST	I-INST	O
Corrected tags	O	O	B-INST	I-INST	I-INST	I-INST
Case 3						
Document	Studied	In	Manipal	Institute	Of	Technology
Predicted tags	O	O	B-INST	I-INST	O	I-INST
Corrected tags	O	O	B-INST	I-INST	I-INST	I-INST

Table 3 Degree name cases covered by correction module

Case 1						
Document	Bachelor	Of	Technology	from	2015	2019
Predicted tags	B-DEG	I-DEG	I-DEG	I-DEG	O	O
Corrected tags	B-DEG	I-DEG	I-DEG	O	O	O
Case 2						
Document	BTech	in	IT	from	2015	2019
Predicted tags	B-DEG	I-DEG	O	O	O	O
Corrected tags	U-DEG	O	O	O	O	O
Case 3						
Document	Studied	In	Manipal	Institute	Of	Technology
Predicted tags	O	O	B-INST	I-INST	B-DEG	I-DEG
Corrected tags	O	O	B-INST	I-INST	I-INST	I-INST

Some of the cases covered by the correction module for institute name and degree is given in Tables 2 and 3, respectively. In case 3 of Table 3 the term 'of technology' is a common occurrence in degree names (Bachelor 'of technology') as well as institute names. The model hence predicted it as I-DEG but correction module rectified it by associating the term with its neighboring words and finding the name Manipal Institute of Technology in the institute list created by us.

2.4.3 Corpus expansion and retraining

The newly annotated data produced by our correction module, for a set, is added to the training set of our corpus. The model is retrained over the newly formed training set for future predictions. This addition to the corpus is repeated for each set of unlabeled data until all the data is exhausted. After retraining the model every time, it is tested against the test data we had set aside initially during the train-test split to ascertain the result's accuracy.

3 Experimental results and discussion

The performance of our model is based on four parameters, namely accuracy, precision, recall, and F1 score [22]. The precision is the percentage of correctly identified entities out of the total identified entities, and the recall is the percentage of entities present in the dataset found by our model. F1 is the harmonic mean of precision and recall. Accuracy is the number of correct predictions upon the total number of predictions.

The experimental result is evaluated based on the evaluation script for the Conference on Computational Natural Language Learning (CoNLL) 2003 shared task [33]. The size of the training set initially (seed) is 500 resume's education sections, and the test set comprises of education sections of 50 resumes. The result obtained are given in Table 4, 5, 6, and 7, respectively. Initially, the program counted 1574 tokens (words and punctuation

Table 4 Performance for institute names

Iteration	Precision (%)	Recall (%)	F1 score
0 (Seed)	39.19	52.25	44.79
1	51.16	59.46	55.00
4	51.20	57.66	54.24
8	56.39	67.57	61.48
12	64.52	72.07	68.09

Table 5 Performance for degree names

Iteration	Precision (%)	Recall (%)	F1 score
0 (Seed)	65.12	73.68	69.14
1	70.23	80.70	75.10
4	81.03	82.46	81.74
8	75.20	82.46	78.66
12	78.26	78.95	78.60

Table 6 Combined performance for degree and institute names with CRF layer included in the model

Iteration	Accuracy (%)	Precision (%)	Recall (%)	F1 score
0 (Seed)	82.08	49.06	57.78	53.06
1	86.09	58.70	64.44	61.44
4	83.42	49.45	60.44	54.40
8	87.87	62.71	65.78	64.21
12	87.99	69.13	70.67	69.89

Table 7 Combined performance for degree and institute names with proposed model

Iteration	Accuracy (%)	Precision (%)	Recall (%)	F1 score
0 (Seed)	85.71	51.26	63.11	56.57
1	89.14	60.77	70.22	65.15
4	90.15	65.56	70.22	67.81
8	90.15	65.50	75.11	69.98
12	92.06	71.13	75.56	73.28

signs) with 225 phrases according to the corpus. Our model found 263 phrases, of which 148 were correct. Note that in Table 4, 5, 6, and 7, the iteration number is equal to the number of unlabeled sets added to the seed set.

With every iteration, each set of unlabeled education sections is fed to the classification and correction module. As the results in Tables 4, 5, and 7 suggest, the increase in training corpus size leads to increased model performance on test data.

It is known that CRF performs better with structured data [5], and since resumes are semi-structured documents, we have experimented with the CRF layer, and the results are given in Table 6. The results obtained through our proposed model are given in Table 7. Our model excludes the CRF layer, and from Tables 6 and 7, we can infer that the proposed model gives more accurate results while being less complex.

To present the effectiveness of our proposed model, we ran the model without the correction module on 860 (500 + 360) manually annotated resumes. The performance of this supervised model (see Table 8) is similar to the performance achieved by our semi-supervised proposed model (see Table 7), which lacked annotated data. It solidifies our claim that our proposed model helps overcome the paucity of large annotated data.

3.1 Comparison with other approaches

Note that due to limited work on resumes for the NER task and the absence of a standard and large dataset (due to the proprietary nature of the data), a direct comparison between any two techniques presented in different studies may not be viable or fair. We have proposed a novel technique and would like to draw a parallel for the same by comparing it with the available literature on NER tasks for resumes. Our work focuses on the education section, since the degree and institution that the degree is obtained from would be available in only the education section of the resume. It may be challenging to directly imply our technique's superiority due to the factors mentioned above. In this section, we have provided a relative performance comparison and not direct. The results compared with other approaches are purely based on the overall accuracy and precision mentioned therein.

Table 8 Combined performance without the correction module: supervised learning

Accuracy (%)	Precision (%)	Recall (%)	F1 score
89	74	77	75

The literature on resume parsing using a semi-supervised NER approach is limited. Zhang et al. [8] used resume document block analysis based on pattern matching, multi-level information identification, and feedback control algorithms and obtained an accuracy of 81% on complex items (like school name) using a larger dataset (2000 resumes for training and 3000 resumes for testing), all manually tagged. Ayishathahira et al. [1] made use of the Bi-LSTM-CNN model to arrive at an F1 score of 76 and 73 for qualification and institution names, respectively, with manual tags. Our model achieved an F1 score of 68.09 for institute names, an F1 score of 73.28 for institute and degree names together, and total accuracy of 92.06% with only 500 handcrafted resumes and 360 automatically tagged resumes.

The cascaded hybrid model described by Yu et al. [36] gives an average *F*1 score of 73.20 for degrees, but for graduate school, the average *F*1 score is only 40.96 which is considerably lower than the proposed model. The graph-based semi-supervised approach proposed by Zafrani et al. [37] for NER can achieve an F1 score of 41.51 for organizations.

For a recruiter, it is essential to scrutinize and filter out candidates based on their resume. Since resumes contain much information about the candidate, a fast way of parsing through the resume and obtaining relevant information is essential. It can be achieved by the process of identifying degree and institute names - to gauge the educational qualifications. Previous work done in this field provide low precision and recall for institute and degree name recognition [34, 37]. Our proposed work can provide improved performance in identifying these entities without the need for mostly annotated resumes, which is generally required for the NER task [19, 20, 37]. The correction module substitutes for the manual annotation, assuming that the initial predictions made by the model are reliable, by rectifying those predictions. The experimental results are in coherence with the above claim, and we see an increase in overall accuracy.

Due to the privacy issues about the personal details present in resumes, they are not available in large quantities for research. Therefore, the identification of the peak of accuracy could not be achieved. Although these constraints exist, the given amount of iterations indicates increasing accuracy with the increase in the availability of resumes.

We have tried to prove that one can achieve high performance for the NER task using our technique even if enough annotated resumes are not available. We have been able to show that even though there may be a paucity of annotated resumes, we can achieve high performance for the NER task with unannotated resumes itself.

4 Conclusion

In this paper, we presented a model for accurate identification of degrees and institute names in a resume based on NER. It is composed of Word Embeddings, CNN, and Bi-LSTM. A correction mechanism for rectification of the predictions made on the unlabeled data is incorporated. These corrected predictions are added to the training corpus. As the neural network processes more unlabeled data, the model accuracy on retraining increases. It achieves high performance without the need for extensive annotated data using a semi-supervised approach. An overall F1 score of 73.28 and an accuracy of 92.06% are obtained. Future work in this area includes extending this model for majors, specialization, and other components of an education section.

Acknowledgements We want to thank Kartik Mandaville and Anshaj Goel from SpringRole India Private Limited for their insight and expertise that greatly assisted our research work. We thank the anonymous reviewers whose insightful comments and suggestions have significantly improved this paper.

Funding Open access funding provided by Manipal Academy of Higher Education, Manipal.

Availability of data and material A private company owns the dataset used for this research work, and due to privacy issues, it cannot be made public.

Compliance with ethical standards

Conflicts of interest Authors do not have any conflict of interest or competing interest to declare.

Code availability Code is not available on any public server; however, it can be provided if the need arises.

References

1. Ayishathahira C, Sreejith C, Raseek C (2018) Combination of neural networks and conditional random fields for efficient resume parsing. In: 2018 International CET conference on control, communication, and computing, IC4 2018, pp 388–393. Kerala, India. https://doi.org/10.1109/CETIC4.2018.8530883
2. Babar N (2017) The Levenshtein algorithm–Cuelogic,. Available: January 25, 2017. https://www.cuelogic.com/blog/the-levenshtein-algorithm. (Online; accessed 7-August-2019)
3. Bird S (2018) Natural language toolkit (nltk). https://github.com/nltk/nltk. Commit: 199c30c5cb5dbb46f5931c9d5b926617bc6a588f
4. Bird S, Klein E, Loper E (2009) Natural language processing with python: analyzing text with the natural language toolkit. O'Reilly Media, Sebastopol
5. Burr T, Skurikhin A (2015) Conditional random fields for pattern recognition applied to structured data. Algorithms 8(3):466–483. https://doi.org/10.3390/a8030466
6. Chifu ES, Chifu VR, Popa I, Salomie I (2017) A system for detecting professional skills from resumes written in natural language. In: 2017 13th IEEE international conference on intelligent computer communication and processing (ICCP), pp 189–196. https://doi.org/10.1109/ICCP.2017.8117003
7. Chollet, F.: Keras: Deep learning for humans. https://github.com/keras-team/keras (2018). Commit:bf1378f39d02b7d0b53ece5458f9275ac8208046
8. Chuang Z, Ming W, Guang LC, Bo X, Zhi-qing L (2009) Resume parser: Semi-structured chinese document analysis. In: 2009 WRI world congress on computer science and information engineering, vol 5, pp 12–16. https://doi.org/10.1109/CSIE.2009.562
9. Darshan PM (2017) Ontology based information extraction from resume. In: 2017 international conference on trends in electronics and informatics (ICEI), pp 43–47. https://doi.org/10.1109/ICOEI.2017.8300962
10. Farkas R, Dobó A, Kurai Z, Miklós I, Nagy Á, Vincze V, Zsibrita J (2014) Information extraction from hungarian, english and german cvs for a career portal. In: Prasath R, O'Reilly P, Kathirvalavakumar T (eds) Mining intelligence and knowledge exploration. Springer, Cham, pp 333–341
11. Ghufran M, Bennacer N, Quercini G (2017) Wikipedia-based extraction of key information from resumes. In: 2017 11th international conference on research challenges in information science (RCIS), pp 135–145. https://doi.org/10.1109/RCIS.2017.7956530
12. Gonzalez J (2018) Fuzzywuzzy: Fuzzy string matching in python. https://github.com/seatgeek/fuzzywuzzy. Commit: 3f820e657c019cbaad964dceee48c34b80c88ba1
13. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. The MIT Press, Cambridge
14. Graves A, Mohamed AR, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing, pp 6645–6649. https://doi.org/10.1109/ICASSP.2013.6638947
15. Jacob F, Javed F, Zhao M, Mcnair M (2014) scool: A system for academic institution name normalization. In: 2014 international conference on collaboration technologies and systems (CTS), pp 86–93. https://doi.org/10.1109/CTS.2014.6867547

16. Jiang Z, Zhang C, Xiao B, Lin Z (2009) Research and implementation of intelligent chinese resume parsing. In: 2009 WRI international conference on communications and mobile computing, vol 3, pp 588–593. https://doi.org/10.1109/CMC.2009.253
17. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP (2016) On large-batch training for deep learning: Generalization gap and sharp minima. CoRR abs/1609.04836, pp 1–16. arXiv:1609.04836
18. Khanam MH, Khudhus MA, Babu MSP (2016) Named entity recognition using machine learning techniques for telugu language. In: 2016 7th IEEE international conference on software engineering and service science (ICSESS), pp 940–944. https://doi.org/10.1109/ICSESS.2016.7883220
19. Lample G, Ballesteros M, Subramanian S, Kawakami K, Dyer C (2016) Neural architectures for named entity recognition. In: Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies, Association for Computational Linguistics, pp 260–270. https://doi.org/10.18653/v1/N16-1030. http://www.aclweb.org/anthology/N16-1030
20. Lee JY, Dernoncourt F, Szolovits P (2017) Transfer learning for named-entity recognition with neural networks. CoRR abs/1705.06273, pp 1–5. arXiv:1705.06273
21. Maheshwary S, Misra H (2018) Matching resumes to jobs via deep siamese network. In: Companion proceedings of the the web conference 2018, WWW '18, pp 87–88. International world wide web conferences steering committee, republic and canton of Geneva, Switzerland. https://doi.org/10.1145/3184558.3186942
22. Manning C, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press, Cambridge
23. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. CoRR abs/1301.3781, pp 1–12. arXiv:1301.3781
24. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th international conference on neural information processing systems, Vol 2, NIPS'13, pp 3111–3119. Curran Associates Inc., USA
25. Pawar S, Srivastava R, Palshikar GK (2012) Automatic gazette creation for named entity recognition and application to resume processing. In: Proceedings of the 5th ACM compute conference: intelligent & scalable system technologies, COMPUTE '12, pp 15:1–15:7. ACM, New York. https://doi.org/10.1145/2459118.2459133
26. Pham T, Le-Hong P (2017) The importance of automatic syntactic features in vietnamese named entity recognition. CoRR abs/1705.10610, pp 1–7. arXiv:1705.10610
27. Pham TH, Le-Hong P (2018) End-to-end recurrent neural network models for vietnamese named entity recognition: word-level versus character-level. In: Hasida K, Pa WP (eds) Computational linguistics. Springer, Singapore, pp 219–232
28. Ratinov L, Roth D (2009) Design challenges and misconceptions in named entity recognition. In: Proceedings of the thirteenth conference on computational natural language learning, Association for computational linguistics, pp 147–155
29. Ravindranath VK, Deshpande D, Girish KVV, Patel D, Jambhekar N (2019) Inferring structure and meaning of semi-structured documents by using a gibbs sampling based approach. In: 2019 international conference on document analysis and recognition workshops. Proceedings of the international conference on document analysis and recognition. Australia, Sydney, pp 169–174
30. Sayfullina L, Malmi E, Liao Y, Jung A (2017) Domain adaptation for resume classification using convolutional neural networks. In: International conference on analysis of images, social networks and texts, Springer, pp 82–93
31. Singh A, Rose C, Visweswariah K, Chenthamarakshan V, Kambhatla N (2010) Prospect: A system for screening candidates for recruitment. In: Proceedings of the 19th ACM international conference on information and knowledge management, CIKM '10, pp 659–668. ACM, New York. https://doi.org/10.1145/1871437.1871523
32. Tikhonova M, Gavrishchuk A (2019) NLP methods for automatic candidate's cv segmentation. In: 2019 international conference on engineering and telecommunication, EnT, Moscow, Russia, pp 1–5. https://doi.org/10.1109/EnT47717.2019.9030535
33. Tjong Kim Sang EF, De Meulder F (2003) Introduction to the conll-2003 shared task: Language-independent named entity recognition. In: Proceedings of the seventh conference on natural language learning at HLT-NAACL 2003, Vol 4, CONLL '03. Association for Computational Linguistics, Stroudsburg, PA, USA, pp 142–147. https://doi.org/10.3115/1119176.1119195
34. Tran Q, MacKinlay A, Jimeno-Yepes A (2017) Named entity recognition with stack residual LSTM and trainable bias decoding. CoRR abs/1706.07598, pp 1–10. arXiv:1706.07598
35. Xu Z, Burdick D, Raschid L (2016) Exploiting lists of names for named entity identification of financial institutions from unstructured documents. CoRR abs/1602.04427, pp 1–18. arXiv:1602.04427
36. Yu K, Guan G, Zhou M (2005) Resume information extraction with cascaded hybrid model. In: Proceedings of the 43rd annual meeting on association for computational linguistics, ACL '05. Association for Computational Linguistics, Stroudsburg, PA, USA, pp 499–506. https://doi.org/10.3115/1219840.1219902
37. Zafarian A, Rokni A, Khadivi S, Ghiasifard S (2015) Semi-supervised learning for named entity recognition using weakly labeled training data. In: 2015 the international symposium on artificial intelligence and signal processing (AISP), pp 129–135. https://doi.org/10.1109/AISP.2015.7123504
38. Zhang C, Wang H, Wu Y (2017) Resumevis: A visual analytics system to discover semantic information in semi-structured resume data. pp 1–9. CoRR arXiv:1705.05206
39. Zhang M (2017) Pdf2json. https://github.com/modesty/pdf2json. Commit:c7d683697e3b69aa84d7acb13d67135cb302d385