2018 IEEE International Conference on Data Mining Workshops (ICDMW)

Extracting Addresses from Unstructured Text using Bi-directional Recurrent Neural Networks

Shivin Srivastava
Dept. of Computer Science and Information Systems
Birla Institute of Technology and Sciences
Pilani, India
h2013073@pilani.bits-pilani.ac.in

***Abstract*—Addresses can be classified as unstructured text because they lack meta-information to be directly indexed in databases. Still they demonstrate an internal structure which can used to automatically extract them using machine learning techniques. In this work we describe a machine learning approach to identify addresses in unstructured text (like blogs) using Bi-directional Recurrent Neural Networks (BRNNs). We overcome the problem of lack of training data by generating synthetic free text entries and come up with problem specific features. Our system [1] does not impose any strict condition on the structure or style of addresses leading to many applications in real life.**

***Index Terms*—Address Extraction; Bi-Directional RNNs;**

I. Introduction

The internet is replete with web sites consisting of primarily, unstructured text. Apart from official uses, the internet is also a popular place for sharing real-life experiences. Food blogs, Travelogues etc. often contain authentic and authoritative information about famous places in a city. All such blogs, regardless of their type, talk about a place(s) of interest, give a short description along with some images and in the end provide an address. Organizing such sources of information be extracting relevant information can be very helpful for various applications like travel itinerary managers, review websites, mapping applications etc.

In this work we focus on the problem of identifying address data from unstructured text like blogs, news etc. Address data is special because it displays a loose structure within itself in the form of components like street names, city, state, phone number, zip code etc. In informal text sources usually found on the internet, the authors do not bother with giving a properly structured address like that in postal cards or business documents. At the same time people have different styles of writing in different parts of the world. This makes the task of tagging addresses even more challenging. Recurrent Neural Networks have proven to be very effective in various NLP applications. Rather, most of the state-of-the-art solutions to problems like text translation, POS tagging etc. are provided by RNNs. In this work we give the first application (as far as we know) of RNNs to the problem of address extraction.

II. Related Work

Before [2] all approaches to address segmentation used hand-coded rule-based methods coupled with a database of cities, states, zip codes etc. [2] gave the first solution to automatically extracting structure from free text addresses using Hidden Markov Models (HMMs). Although their approach managed to extract structure from addresses with a high precision (88.9%) and recall (88.9%) on student addresses, but still it works only with a collection of pure address data. Such a solution will not be able identify and extract address data from unstructured text sources like news or blog data.

[3] present a statistical approach using Conditional Random Fields (CRFs). CRFs are a generalization of HMMs which output confidence measure of the output prediction. The authors trained their model on 400 Web sites manually annotated with address information. They achieved an average precision of 0.89 and an average recall of 0.64 for the single attributes.

The above approaches are competent in the scenarios they describe but none is fit for extracting general addresses from informal, freely written content like news or blogs. In this work we will describe a robust model based on BRNNs which can extract addresses of various formats. Also, the above works have used manually annotated datasets for training. Since RNNs need a large amount of training data, we manually synthesize our own training data to imitate blog posts found on the internet.

III. Structure of Addresses

Since we segment a web page into sentences, our task reduces to the problem of classifying a sentence as an address or non-address type. An address has many components like House/Shop name, Street name, city, country, zipcode, phone number etc. All of the above features may or may not be present in addresses which are not written according to any convention. We classify addresses according to the placement of these components in the address structure.

A. One-line Addresses

Many people write the complete addresses in a single line eg. `ArtBar, Monday, July 4, 40 Edwin Land Blvd., Cambridge, 617-806-4122,artbarcambridge.com.` from Boston Magazine. These type of addresses are relatively easier to tag.

[1] [1] has the full code for the Travello project.

B. Multi-lined addresses

Multi-lined addresses are more common as compared to single line addresses. The formal order writing an address is to mention the **Company name**, **Street Number**, **City**, **County**, **Postal Code** and **Country** but seldom do we find that such a format is followed on the internet. A typical multi-line address may look like this:-

```
6oz Espresso Bar
20 McCallum Street, #01-K1,
Singapore 069046
Tel: +65 6509 6602
Mon to Fri: 7am  5.30pm
Nearest Station: Tanjong Pagar
```

from ladyironchef, a Singapore based food blog

So, although the address might not might not contain all the elements components that must be formally present, still we observe that the ordering between the component remains unchanged ie. Street Number comes after City name etc. Thus we can treat the multi-line addresses as a time series data and use Recurrent Neural Networks to perform a sequence tagging operation.

IV. Methodology

A. Generating the Dataset

Deep Neural Network generally require a large dataset [4] for effective training. Since it is not feasible to manually tag so many addresses in blogs for training, we generate our own training data to imitate a typical blog post which describes multiple places of interest as follows:-

- Select a random number of English sentences taken from a repository (eg. a book) distributed normally with a user specified mean and standard deviation.
- Create a synthetic address as follows and append it to the running text:-
  - First generate the address template, randomly selecting components with fixed probabilities (since an address may not necessarily have all the above described components eg. Street name, City etc.)
  - Populate the template with with random entries chosen from the database to form the imitation address.
- Repeat until required

We use the Walmart-full address dataset [5] as our address repository. By randomly dropping address components in the template, we emulate addresses found in real-life data sources. This makes the classifier more robust to recognizing different address formats which may arise in real life situations. For one-line addresses, we append the address in the same line instead of going to the next line as in hierarchical addresses.

B. Features

Since address components like Street names, city names, state names, country names etc. are not arbitrary, we decided to make a database of all these components. OpenStreetMap provides a collection of all addresses in the United States and the world. We extract all the street names from these addresses to make a custom database. Using regular expressions, we identify phone numbers of various countries. A weight is assigned to every address component depending on frequency of occurrence in its respective database. Presence of phone numbers is indicated by a binary variable. We append all the weights of various address components to form a 9-dimensional feature vector for a sentence.

C. Network Architecture

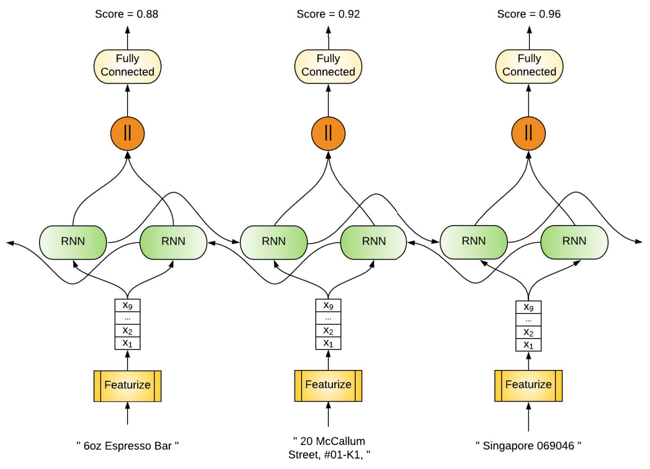

Fig. 1: The Model

We use BRNNs as they can understand the context by observing the past and future simultaneously. In our application this is specifically important as addresses are generally preceded and succeeded by non-address sentences and within addresses also we find a hierarchical structure as described above. The best performance is achieved by a simple model having one forward and one backward layer. After concatenating their results and capping with a $tanh$ dense layer on top, we get favourable results.

D. Identifying Address Sentences

Our model returns a score associated with a sentence which describes how likely it is an address, in out experiments we found that it is not always possible to fix a hard threshold value to demarcate between an address and a non-address sentence. To mitigate this problem we cluster the sentences into two groups using standard clustering algorithms based on the scores assigned by the model. Thus all address sentences are clustered in one group while the non-address sentences are separated to another.

V. Conclusions

We present an approach to extract addresses from unstructured, free text like blogs using Bi-directional Recurrent Neural Networks. We also describe the data generation process along with the suitable features to train the model on.

VI. Acknowledgements

This research is partially funded by National University of Singapore (NUS).

References

[1] (2017) Travello-nlp. [Online]. Available: https://github.com/shivin9/Travello-NLP

[2] V. R. Borkar, K. Deshmukh, and S. Sarawagi, "Automatically extracting structure from free text addresses," *IEEE Data Eng. Bull.*, vol. 23, no. 4, pp. 27–32, 2000.

[3] B. Loos and C. Biemann, "Supporting web-based address extraction with unsupervised tagging," in *Data Analysis, Machine Learning and Applications*. Springer, 2008, pp. 577–584.

[4] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[5] (2013) Upper bound for faulhaber's sum. Last accessed 30 Aug. 2018. [Online]. Available: https://old.datahub.io/dataset/walmart-dataset