Named Entity Recognition for Resumes

Tomoyuki Jinno

Introduction

What is Named Entity Recognition?

NER is a task in natural language processing, where the aim is to classify a words in text into predefined categories.

Fig.0: E.g. of an output of a NER

E.g. When Sebastian NORP Thrun started working on self-driving cars at Google ORG in 2007 DATE , few people outside of the company took him seriously.

AIM

- MeVitae is a company that provides services for automatically shortlisting CVs for organisations.
- There was a need for a NER to extract the key features of a resume for scoring.
- Therefore the aim was to **write a NER program that classify, and extract words required for scoring CVs**. E.g. University names and courses

Why Not Use Pre-Existing NER?

- Publicly available named entity recognition programs are trained on commonly found text corpus such as news articles.
- Sentences found in resumes have different structure to theses. Hence, publicly available NERs did not perform well on resumes.

Method

Neural Network Based Method

- The NER was made by training a **bidirectional LSTM** model.
- LSTM is a recurrent neural network. When trained, it can output sequential predictions from sequential input.
- The network takes a sequence of **part of speech** as an input and return a sequence of predefined class as an output.

E.g.

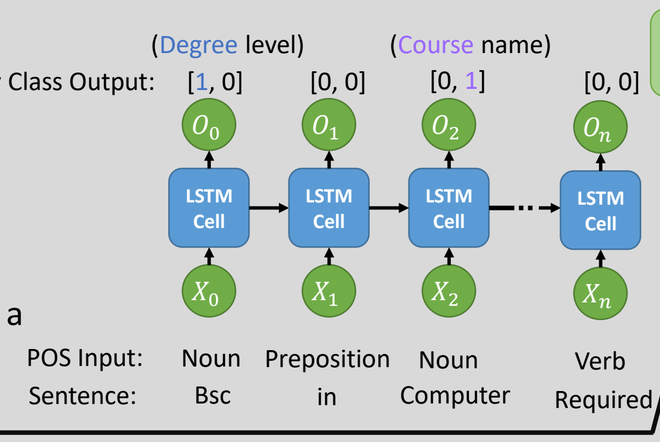

Fig.1: A diagram showing how a LSTM model can be used for NER tasks.

Generating Training Data

- Automated approach was used to label the training data to save time.
- The program labels sentences from CVs using a predefined set of words that belongs to a class.
- ***Problem***: **resulted in many false positives and false negatives in the generated training data.**

The Causes of FPs and FNs:

1. **False negatives:** The set of words do not contain all words that should belongs to the class
2. **False positives:** Context in use of the word is not taken into account (E.g. synonyms)

E.g.

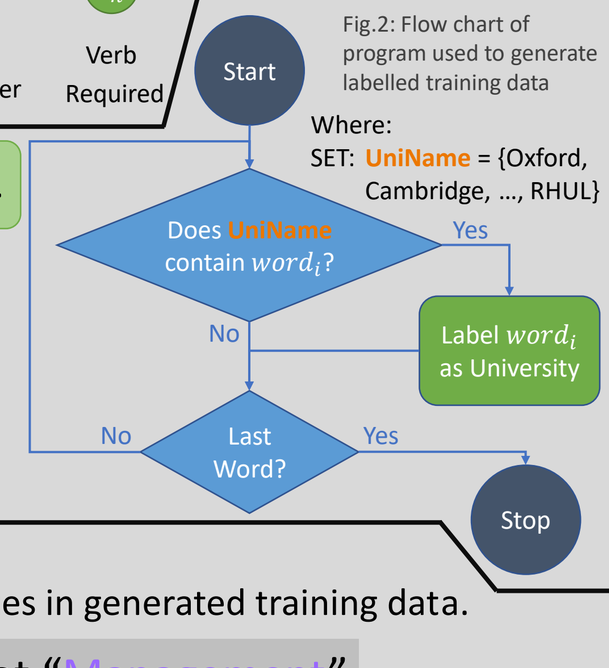

Fig.2: Flow chart of program used to generate labelled training data

Use of Part of Speech as Input for the Model

We used part of speech of the sentence as an input for the RNN. This was done to counter the issues in generated training data.

E.g. If a set of words, "Course names" contained the word "Physics" but not "Management"

	Sentence 1			
Words	Bachelor's	degree	in	Physics
Automatically Generated Labels	0	0	0	1
RNN Input	PROPN	NOUN	ADP	PROPN
RNN Output	0	0	0	1

	Sentence 2			
Words	Bachelor's	degree	in	Management
Automatically Generated Labels	0	0	0	0
RNN Input	PROPN	NOUN	ADP	PROPN
RNN Output	0	0	0	1

True Positive

Identical
(Indistinguishable)

False Negative

Correct Output

Why would using POS as input solve the problem?

If we used words as an input, e.g. word vectors, then the model will learn to make the same mistakes made on the generated labelled training data.

E.g. For Sentence 2, the model would memorise the word "Management" and incorrectly predict it as 0.

If POS is used as an input instead, then the incorrectly labelled sentence and correctly labelled sentence becomes indistinguishable, provided that they have the same sentence structure.

The cases where a sentence structure is labelled incorrectly is a minority, hence the model can predict the incorrectly labelled sentences correctly. Since the model learns to make predictions using sentence structures and not words.

It can not tell apart words that did or did not belong to the set of words used to generate the labelled training data.

Result

- We successfully developed a NER tagger using a bidirectional LSTM.
- Labelled data was generated automatically, but contained false positives and false negatives.
- As shown on Fig.3, our LSTM model was able to give correct predictions, even when the labelled data is incorrect.
- This was achieved by using POS as input instead of word vectors.

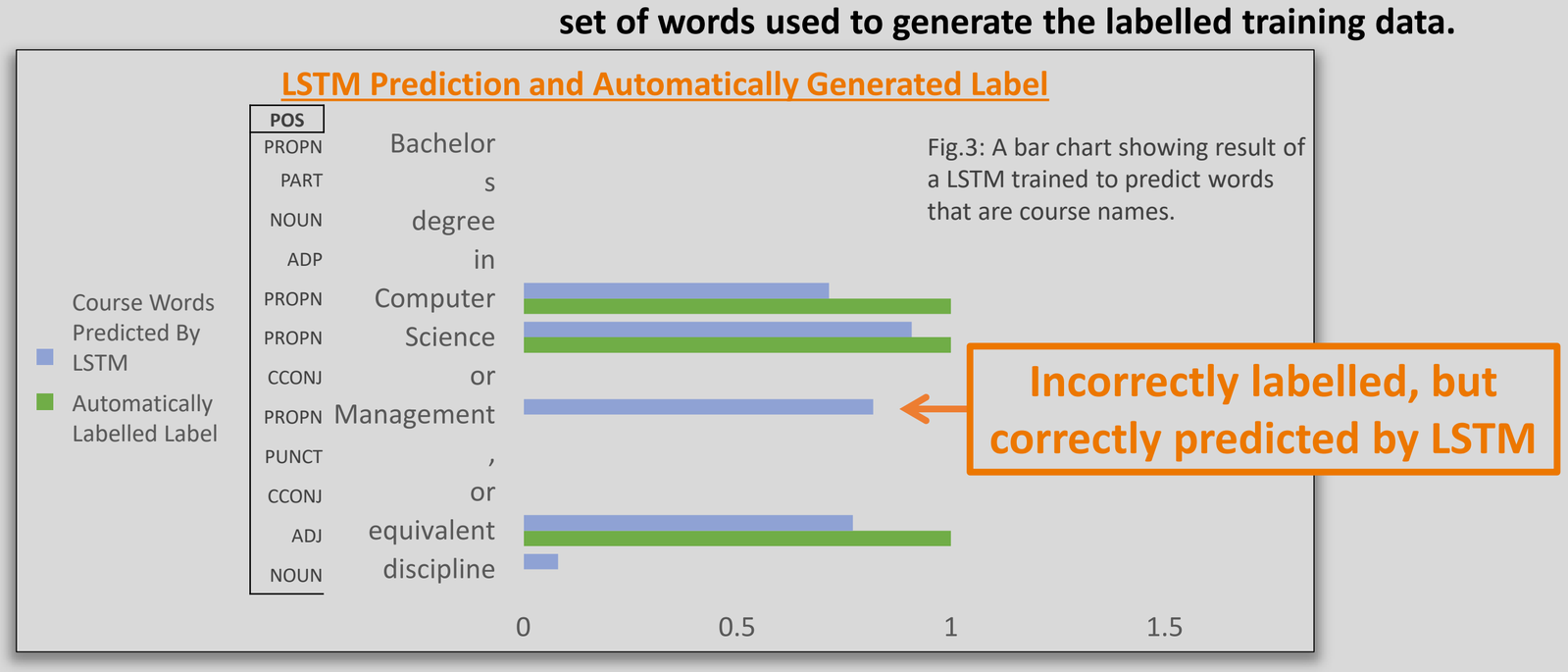

Fig.3: A bar chart showing result of a LSTM trained to predict words that are course names.