2020

Poincaré Embeddings in the Task of Named Entity Recognition

David Muñoz
Fernando Pérez Téllez
David Pinto

Follow this and additional works at: https://arrow.tudublin.ie/ittscicon

Part of the Computer Sciences Commons

This Conference Paper is brought to you for free and open access by the School of Science and Computing at ARROW@TU Dublin. It has been accepted for inclusion in Conference Papers by an authorized administrator of ARROW@TU Dublin. For more information, please contact arrow.admin@tudublin.ie, aisling.coyne@tudublin.ie, gerard.connolly@tudublin.ie.

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 4.0 License
Poincaré Embeddings in the Task of Named Entity Recognition

David Muñoz\textsuperscript{1,2}, Fernando Pérez\textsuperscript{1}, and David Pinto\textsuperscript{2}

\textsuperscript{1} Department of Computing, Technological University Dublin, Blessington Rd, Tallaght, Dublin D24 FKT9, Ireland
devasc26@gmail.com, fernandopt@gmail.com

\textsuperscript{2} Faculty of Computer Science, Benemerita Universidad Autonoma de Puebla, Av. San Claudio, Blvrd 14 Sur, CU, 72592 Puebla, Mexico
davideduardopinto@gmail.com

Abstract. Hyperbolic embeddings have become important in many natural language processing tasks due to their great ability to capture latent hierarchical data and to encode valuable syntactic and semantic information. We study and consider the ability of Poincaré embeddings to get the most similar nodes to a given node when trying to recognize named entities in a set of text documents. In this paper, we propose a classifier model for the NER (Named Entity Recognition) task by implementing Poincaré embeddings and by using the most frequent n-grams and their Part-of-Speech (POS) structures from the training dataset. We found that POS structures and n-grams help to map possible named entities, while using Poincaré embeddings manage to affirm and refine this recognition, improving the recognition of named entities.

Keywords: Poincaré embeddings · NER · Named entity recognition · Part-of-Speech · N-grams

1 Introduction

Nowadays, the amount of textual information available electronically is incredibly abundant, causing an information overload when searching for something specific. It makes difficult to obtain the most relevant information on a specific topic and it is time consuming. To deal with this problem there are information retrieval (IR) systems, which search for information in a collection of documents and retrieve the most relevant resources based on a specific information need [1]. To achieve this some techniques are required such as the recognition of named entities (relevant information) through which a text document can be represented, reducing the information overload [2]. In this sense NER is a subtask of Information Extraction (IE) where the information we are looking for is known beforehand.

In the present work, a NER classifier model is proposed by including Poincaré embeddings in order to get the closest words to a given word. By using the closest words it is possible to know if a tag (class) is near to that given word, and
then assign the tag to that word. But also the use of representative features of each class is considered to improve the correct recognition of the named entities. The named entities that make up each class are converted to n-grams just like their POS structures. Then the most representative n-grams of each class as well as the POS structures are used to refine the correct recognition of the named entities, previously achieved by the Poincare models. It is through this proposal that it is intended to improve the recognition of named entities which can represent a document with the most important information and thus facilitate some tasks such as the classification of texts and retrieval information. To create the Poincaré models we created a Gold Standard by manually tagging a set of documents (Job Descriptions) in the IT field, and then converted to an structured format (tab-separated columns) where the first column contains every token in the document and the second column contains its belonging tag. Then Poincare models can be used for getting vector representations of words and try to recognise the tags of every word.

In summary, this paper presents a classifier model for the task of named entity recognition by employing Poincaré embeddings, n-grams and Part-of-Speech. The rest of the paper is structured as follows. Section 2 describes the background related to our proposal, such as NER Task, Poincaré embeddings, N-grams, and Part-of-Speech. Section 3 explains the design of the proposed classifier model. Section 4 shows the experiments and results carried out, and how Poincaré models, POS structures and n-grams help in the task of NER. Section 5 presents a conclusion of the results obtained.

2 Background

2.1 Named Entities

The Named Entity (NE) task is a subtask of information extraction (IE) and was developed by the committee in the Sixth Message Understanding Conference in 1995. It became an important task in IE field since it aims to locate and classify named entities in a text into categories previously defined (e.g. People, Organizations, Geographic Locations). In this way, texts can be represented by their named entities (most relevant terms) [3]. Another definition for named entity is, roughly speaking, anything that can be referred to with a proper name: a person, a location, an organization. However, the term is commonly extended to include things that are not entities such as, including dates, times, and other kinds of temporal expressions, and even numerical expressions like prices [10].

2.2 Poincaré Embeddings for Word Representation

Hyperbolic geometry is a non-Euclidean geometry which studies spaces of constant negative curvature. It is, for instance, associated with Minkowski space time in special relativity. In network science, hyperbolic spaces have started to receive attention as they are well-suited to model hierarchical data. For instance,
consider the task of embedding a tree into a metric space such that its structure is reflected in the embedding. Due to the underlying hyperbolic geometry, this allows us to learn parsimonious representations of symbolic data by simultaneously capturing hierarchy and similarity [4].

Word embeddings are effective for many natural language processing tasks because they are flexible and encode valuable syntactic and semantic information. Word embeddings are motivated by the concept that semantic similarities between words are based on their distributional properties in the large amount of text. The idea of distributional properties is called distributional hypothesis [5], meaning that linguistic items with similar distributions have similar meanings. Popular word embeddings such as GloVe, Word2Vec, and FastText are widely used in various tasks and have shown great success. Although these embedding methods have proven successful, very few methods exist that are able to encode tree-like or graph-like hierarchical relationships of the data. Poincaré embeddings are better at capturing latent hierarchical information than traditional Euclidean embeddings.

2.3 Part-of-Speech (POS) Tagging

POS tagging is the process of marking a word in a text with a tag carrying the corresponding information to a particular part of speech, based both on its definition and its context (its relation to adjacent and related words in a phrase, sentence or paragraph) [7]. This tagging is done using computational algorithms that associate discrete terms, as well as hidden parts of the speech, through a set of descriptive labels. These POS tagging algorithms are divided into two types: rule-based and stochastic.

2.4 N-gram Language Model

An n-gram is a subsequence of n elements of a sequence of words given in a text and is widely used in the study of natural language. An n-gram model is a type of probabilistic model that allows a statistical prediction to be made of the next element of a certain sequence of elements that has occurred so far. An n-gram model can be defined by a Markov chain of order $n - 1$, due to its relative simplicity to increase the study context by increasing the size of $n$ [6].

3 Proposed Classifier Model

This section explains the characteristics considered in the construction of the classifier model as well as the way in which they were calculated.

3.1 Definition of Used Named Entities

As mentioned in Sect. 2.1 the term Named Entity is commonly extended to include more things that result important for some tasks. In this case we focus
on extracting relevant concepts for our case study where our training set refers to Job Descriptions in the IT field. The classes we found most important are defined as Role, Knowledge, Skill, Talent, Character and Responsibility\footnote{Definitions of used classes are explained in the original source and can be found in https://dictionary.cambridge.org/dictionary/english/}.

- **Role.** The position or purpose that someone or something has in a situation, organization, society or relationship. Some examples are: “Enterprise Technical Support Engineer” and “Senior IT Project Manager”.

- **Knowledge.** Awareness, understanding, or information that has been obtained by experience or study, and that is either in a person’s mind or possessed by people generally. Some examples are: “Understanding of Microsoft Azure” and “Programming languages”.

- **Skill.** A particular ability that you develop through training and experience and that is useful in a job. Some examples are: “Good analytical skills”, “Good troubleshooting skills” and “Excellent communication skills”.

- **Talent.** Someone who has a natural ability to be good at something, especially without being taught. Some examples are: “Enable business capabilities through innovation” and “Self-managing”.

- **Character.** A person, especially when you are describing a particular quality that they have. Some examples are: “Independent” and “Team player”.

- **Responsibility.** The things you are in charge of in your job. Some examples are: “Responsible for providing technical support to customers” and “Responsible for providing after-hours support as part of an on-call rotation”.

### 3.2 N-grams Used in Gold Standard

The corpus is composed of 160 Job Descriptions in the field of IT, collected from the website www.jobs.ie and labelled with the 6 classes previously described. However, we want to know what n-grams are used most in each class. To achieve this we have measured the length of the named entities (n-grams) that make up each class, immediately we grouped them by \( n \) size, and then we counted their frequencies in the corpus. Figure 1 shows the distribution of the existing n-grams in each class.

With the graphic representations of the n-grams can be seen that most of the classes are mainly made up of bigrams, with the Role, Skill and Knowledge classes being the most common. However, the Knowledge and Character classes are mainly represented by unigrams, with a large percentage difference from the rest of the classes. That means that these two classes are characterized by being unigrams and bigramas. On the other hand, it is possible to observe that the trigrams occupy the third position in representing each of the classes, being the only one more balanced group based on the number of occurrences. Four-grams represent from 6\% to 12\% of the composition of each class, ranking fourth in the main named entity sizes. As the rest of the n-grams have such a low presence in the classes, with the exception of the Responsibility class, being the only class...
that represents a less disproportionate distribution of named entities of short and long length. Furthermore, the Responsibility class has the longest n-grams existing in the corpus, being something characteristic of this class.

3.3 POS Structures of N-grams

POS tagging was applied by using the tagger provided by the Natural Language Toolkit library, then all the sequences found in the named entities were grouped (and the occurrences of each one were counted). The default function POS tagset was used because it offers the possibility of tagging with 35 different classes [8], making the structure of a named entity more precise and thus recognizing its similarity with another. In Fig. 2 it can be seen which are the POS-tags of the most representative named entities for each class.

The class **Skill** is mainly made up of bigrams, which is something that characterizes the class and it is worth looking at its component structures: *NN NN, JJ NN, NN NNS, NN VBG, JJ JJ, NN RB, NNP NNP, VBG NNS, JJ NNS and NNP NN*, where nouns and adjectives are the most used POS tags and resulting characteristic of their named entities.
On the other hand, the class *Role* is made up mostly of bigrams and trigrams, so it would be very rare for an named entity in this class to be just one word. The main structures of bigrams in this class are: *NNP NNP, NN NN, JJ NN* and *NN NNP*, while the main structures of trigrams are: *NNP NNP NNP* and *JJ NNP NNP*, being something characteristic of this class that its named entities are mainly made up of proper nouns in the singular form.

In the class *Character* mainly unigrams are used (*NN* and *JJ*), while the most present bigrams have the structure *JJ NN*, and most of the trigrams have the structure *NN TO VB*.

While the class *Knowledge* is strongly represented by unigrams of the *NN* form, followed by the *NNS* form, and bigrams with the structures *NN NNS*,...
The class *Talent* is made up of unigrams, bigrams and trigrams, with the bigrams being the most representative and with the following structures: *VB NNS*, *NN NN* and *JJ NN*. The main trigrams have the structures: *NN TO VB* and *NN NN NNS*, and unigrams are represented with the form *NN*. This class strongly uses nouns, which are part of any of its named entities.

Finally, the class *Responsibility* is mainly made up of trigrams that have the following structures: *NNP NN NNS*, *NN NN NNS*, *VBG NN NNS*, *VBG JJ NN*, *JJ NN NNS*, *VBG JJ NNS*, *NN NN NN*, *VB JJ NNS*, *VBG NN NN* and *NNP JJ NNS*. And those trigrams are formed with singular and plural nouns, adjectives and verbs in gerund (present participle). The main bigrams that conform the class have the structures: *NN NN*, *NN NNS*, *JJ NN*, *VBG NNS* and *NNP NN*, where it is possible to realize that these bigramas are formed in the same way as the trigrams (with nouns, adjectives and verbs in gerund). And as for the unigrams they are characterized for being *NN* and *VBG*, concluding that the use of verbs in present participle is something characteristic of the class *responsibility*.

### 3.4 Poincaré Models

The corpus consist of 121,293 tokens, and a vocabulary of 5,297 unique words. The file format in which the information is stored is double column, saving the relation of a token in the first column to its assigned class in the second column. By using the gensim library was possible to train the Poincaré model with the corpus file. The number of dimensions of the trained model is set to 50 by default and it also needs a number of iterations which is set to 50 by default. As suggested in [4] all embeddings were randomly initialized from the uniform distribution $U(-0.001, 0.001)$, causing embeddings to be initialized close to the origin. The main focus for Poincare embeddings is its capability to embed data that exhibits latent hierarchical structures. Once the model is trained it is possible to find the closest words to a user-specified word through the most_similar method implementation where the top-N most similar nodes to the given node (word) is returned in increasing order of distance.

This characteristic is used to get the most similar words for each one of the words conforming to a named entity. Between the most similar words are also

### Table 1. Assigning classes to NEs using Poincaré embeddings

| Tested NE                           | Original class | Predicted classes                          | Most frequent class |
|-------------------------------------|----------------|--------------------------------------------|--------------------|
| Integration developer               | Role           | Role, Knowledge, Skill, Role, Role         | Role               |
| Providing support                   | Responsibility | Responsibility, Responsibility, Knowledge,  | Responsibility      |
|                                     |                | Skill, Skill                              |                    |
| Excellent communication             | Skill          | Skill, Responsibility, Skill               | Skill              |
included classes belonging to each word. Then the most frequent class in every named entity is selected, discarding others that occur just a few times. In Table 1 are showed some examples for named entities and the closest words returned by the Poincaré model implementation. So, for the named entity Excellent communication the classes Skill and Responsibility are predicted, however the class Skill occurs more times. That is the reason of selecting the class Skill as the assigned class to the named entity which is correct. And the same happens for every one of the named entities, showing excellent results as the predicted class is the original assigned class in the corpus.

3.5 Process of Construction of the Classifier Model

By making use of Poincaré embeddings our classifier intends to improve the recognition of named entities which represent Job Description documents with the most relevant information and thus facilitate some tasks such as the classification of texts and retrieval information. The classifier works considering the features described before when n-grams and part-of-speech were studied over the corpus.

In Sect. 3.2 we found that most of the classes are conformed by unigrams, bigrams and trigrams, being cuatrigrams also present in good percentage. N-grams with five or more elements are unusual, that is why when finding named entities we consider a window size of 4 words in order to find a relation between the current word and the next 4 words.

Fig. 3. Workflow diagram implemented on the classifier model
When studying the representative part of speech structures for each class, some patterns were found. However the most interesting structure was found for the class *Role* where named entities are normally proper nouns in singular form (NNS), and that is something very characteristic of that class. Figure 3 shows a diagram of the workflow operation implemented on the classifier model developed. The process consists of 3 stages for the recognition of named entities.

**Classifier Design**

- **Match Gold Standard named entities in test dataset (Stage 1).** From the original set of named entities in training dataset, if they exist in the test dataset then they are tagged. In other words, it consists in searching the entire list of training named entities in the test set and assign them the corresponding tags.

- **Search POS structures (Stage 2).** In stage 2, first an analysis of the named entities of each class is performed, thus determining the number of most used n-grams in each class, as well as their POS structures. Thus, only those structures with the most occurrences for each class will be considered. Once these most representative features of each class have been selected, they will be searched in the test dataset, and then there will be possible named entities to be tagged.

- **Use Poincaré embeddings and most frequent n-gram sizes (Stage 3).** By employing Poincaré embedding models, with the most frequent n-gram sizes it is verified if the words in that windows size have the same tag as one of the 5 closest neighbors. If all of the words on the window size have the same tag, then they are tagged and considered as a named entity. A more detailed explanation of how this stage works is shown in Fig. 4.

In stage 2, named entity candidates are found considering features of each class, but is in stage 3 that these named entities are tagged or discarded. In stage 3, for each of the words that make up a possible named entity, its closest neighbor is obtained, and if the closest neighbor for any of these words is one of the classes used in the corpus, then this word is tagged with that class and becomes a seed word or root word. Then a window of 4 words on the left and 4 on the right is used, and the 5 closest neighbors are obtained for each of those words and if the same class as the root word is found within the closest neighbors, then this word is added to the root word, and so on until the words inside the window are finished. At the end, a named entity is obtained with the words that share the same tag within the range of the window.

**4 Experiments and Results**

To know how the proposed classifier model developed performs, an experiment is carried out demonstrating the help of the Poincare embeddings when predicting named entities. For this experiment we have used a Gold Standard that we
Fig. 4. Representation of how Poincaré models are used to predict named entities.

have manually tagged with a text annotation tool called Web-based Tagger tool developed in [9]. This annotation tool allows users to highlight the most relevant concepts of a set of text documents, and then the tool transforms this data to an structured format ready to be used in the training of some predictive models. The Gold Standard is composed of a set of 160 Job Description documents in the field of IT and was collected from the website www.jobs.ie and tagged with the 6 classes described previously.

Table 2. Performance of classifier when using different stages of the workflow process

| Performance | Classifier model | | |
|-------------|------------------|------------------|------------------|------------------|
| Class       | Measure | Stage 1 | Stage 1 & 2 | Stage 1 & 2 & 3 |
| Character   | F-1     | 0.6381  | 0.6710  | 0.8057  |
| Knowledge   | F-1     | 0.5037  | 0.5800  | 0.7681  |
| Responsibility | F-1     | 0.1760  | 0.3250  | 0.4578  |
| Role        | F-1     | 0.4433  | 0.5480  | 0.6553  |
| Skill       | F-1     | 0.5068  | 0.5553  | 0.7046  |
| Talent      | F-1     | 0.4224  | 0.4695  | 0.5719  |

2 The Web-based Tagger Tool can be found in https://lkesymposium.tudublin.ie/Tagger/.
The use of features improves the performance of the proposed classifier thanks to the fact that these characteristics allow named entities to be identified more accurately. To demonstrate the usefulness of these features, we present a series of experiments, where the first experiment only considers the first stage of the workflow diagram, the second experiment considers the first and second stages of the workflow diagram, and the third experiment considers all the stages of the workflow diagram showed in Fig. 3.

V-fold cross validation method was used to measure the classifier’s performance considering different features presented on the workflow diagram, and the averaged obtained results are presented in Table 2. It can be seen that Stage 3 plays an important role in recognizing named entities. The main reason for seeing a notable improvement here is the fact of using Poincaré embeddings, since in stages one and two the most common and highly secure named entities have been tagged, so in stage 3 the amount of untagged information has decreased considerably and thus it is easier to analyze the remaining data and determine through the closest words to a root word potentially identified as belonging to a class.

Figure 5 shows a comparison of the model performance when implementing different stages of the classifier. In stage one all those named entities that exist in the training corpus are tagged, that is why the recognition level is not so high, in addition to that the Precision in this stage is generally high as observed in Table 2. However, the second stage allows POS structures to be recognized based on the POS structures of each class in the training corpus, thus increasing the Recall level and decreasing Precision. Finally, the third stage allows to create a better balance between Precision and Recall, when using the Poincare embeddings since they establish a relationship between a root word with an assigned class and allows to know which words around it belong to the same class and therefore to the same named entity.
5 Conclusions

As it is possible to observe, the use of Poincaré embeddings improves the performance of the proposed classifier thanks to the fact that this feature allows named entities to be identified more accurately by using the closest words to a given word. The word representations obtained by the Poincaré models effectively captured the existing relationships between words, helping us to always identify if for a given word there is a class close to it.

The previous analysis of the POS structures of the n-grams allowed us to know which structures are most representative of each class and, therefore, to take them into consideration to determine possible named entities. In this way, the recognition of named entities could be expanded compared to stage 1.

On the other hand, having carried out a previous analysis of the n-grams that make up each one, could lead us to consider mainly those n-grams that occurred mostly in each class. In this way, it was possible to define a window size of words contiguous to a word to be analyzed, and then be able to know if those words together with the root word can form a named entity. In this way, for each contiguous word, the Poincaré model was called to determine if the same class of the root word was found within the closest words, achieving an improvement in prediction. But the results showed that most of the classifier improvement occurs in stage 3 where Poincaré models are used. The effectiveness of Poincaré embeddings to capture the relationships between words in a context in a hyperbolic space are helpful when we want to recognize the named entities in a text.

References

1. Jansen, B.J., Rieh, S.: The seventeen theoretical constructs of information searching and information retrieval. J. Am. Soc. Inf. Sci. Technol. 61, 1517–1534 (2010)
2. Hasegawa, T., Sekine, S., Grishman, R.: Discovering relations among named entities from large corpora. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, p. 415. Association for Computational Linguistics (2004)
3. Grishman, R., Sundheim, B.: Design of the MUC-6 evaluation. In: Proceedings Sixth Message Understanding Conference (MUC-6), pp. 1–11 (1996)
4. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: Advances in Neural Information Processing Systems, pp. 6338–6347 (2017)
5. Harris, Z.S.: Distributional structure. Word 10(2–3), 146–162 (1954)
6. Lay, D.C.: Linear Algebra and Its Applications, 5th edn, pp. 255–262. Addison Wesley Publishing Company, Boston (2018)
7. Sketch Engine: POS tagger (2020). https://www.sketchengine.eu/my_keywords/pos-tagger/
8. Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python. O’Reilly Media, Sebastopol (2009)
9. Muñoz, D., Pérez-Téllez, F., Pinto, D.: Collaborative web-based tagger for named entities in the task of information extraction. Pistas Educativas 40, 877–893 (2018)
10. Jurafsky, D., Martin, J.H.: Information extraction. In: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, pp. 725–743 (2009)