Phrase Extraction models used for Entity Salience Detection

Radboud Universiteit Nijmegen

Master Thesis Computing Science
January 26, 2019

Author:
Kevin Jacobs

Supervisor:
Prof. dr. ir. Arjen P. de Vries

Abstract

In this thesis, entity salience detection is researched and the publicly available WikiNews dataset is augmented with phrases information per document-entity pair. The phrases are then used together with positional encodings to determine the salience of a given entity for a document-entity pair. The results are compared to the frequently used positional baseline for entity salience and pointers are given for future research directions. Besides that, an end-to-end differentiable phrase extraction and entity salience detection model is implemented and directions for improving this model are given.

Acknowledgements

I would like to thank my supervisor, Prof. dr. ir. Arjen P. de Vries for his advice, positive attitude and his time he has provided me during the research. Furthermore, I would like to thank both my family and my girlfriend for their support, patience and encouragement.

Contents

Chapter 1

Introduction

1.1 Motivation

"What is this text about?" is a hard question for computers without a trivial answer. Text summarization is a hard NLP task which is related to the former question. One could also ask the following question: "About which entity (or entities) is this text?". The task of finding these entities is called entity salience detection. An answer to this question could help to improve on text summarization.
Besides the fact that the entity salience task could be useful for text summarization, it is also a collection of entity-based Natural Language Processing (NLP) tasks. Methods for Named Entity Recognition (NER) and Entity Linking (EL) are frequently used in solutions to the entity salience task.
This task is interesting since it sheds light on Natural Language Understanding, which is an active research area on how machines could understand a text.
The main focus of this research is the contribution of external knowledge for achieving higher performance on the entity salience task.

1.2 Research question

The entity salience classification task is the following. Given a text document and the name of an entity, determine the importance (salience) of the entity within the document. In the classification task, an entity is either salient or non-salient. The following research question is answered in this thesis: What is a good way to use properties of the text in order to detect the most important entities in a text?

1.3 What are the main contributions of this research?

This work provides a new dataset, the Wikiphrase dataset, which is an enrichment of the Wikinews dataset found in [19]. The notebooks that were created and used for the preprocessing and processing of the datasets are publicly available on GitHub. Furthermore, a research direction is given for entity salience detection. Phrase extraction might help to increase the recall for this task. Also an end-to-end differentiable model for both phrase extraction and entity salience detection is implemented (mESD) and future research directions in order to improve this system are given.

1.4 Methods

In this work, the entity salience task is viewed as two subtasks. The task of entity phrase detection and the task of entity salience classification. The classification task is based on the result of the entity phrase detection task. The errors made in the phrase extraction task might degrade the precision and recall of the entity salience classification method. The chapter on phrase extraction introduces several phrase extraction methods. The chapter on entity salience classification describes a method based on the results of a phrase extraction method. The chapter on both tasks introduces the mESD (mLSTM Entity Salience Detection) model for both for phrase extraction and entity salience detection.

In this work, a dataset is created. The process of creating this dataset is described in the chapter about the datasets.

Chapter 2

Related work

In this chapter, the related work is described. First, research on word embeddings is described. Then, the entity salience task is described and after that, research on tasks related to the entity salience task is described.

2.1 Word embeddings

Word embeddings are numerical representations of words. Most of the recent research in NLP incorporate word embeddings. Some word embeddings describe the sequence of characters of a word (syntactic word embeddings) and other word embeddings encode semantic information related to the word (semantic word embeddings). Frequently used word embeddings are GloVe embeddings [14] and Word2Vec embeddings [13]. In this thesis, the GloVe embeddings [14] are used. The pretrained word embeddings encapsulate linear relationships between words. For example, let $w(x)$ represent the GloVe vector for a word x. A classical example of a linear relationship that holds in the pretrained GloVe vectors is the following: $w(\text{king}) - w(\text{man}) + w(\text{woman}) \approx w(\text{queen})$.

2.2 Entity salience task

Entities are used in numerous NLP tasks. In this work, the entity salience task is discussed. In the entity salience task, the most salient entities for a given document are extracted. This work is mainly inspired by the work of Dunietz [3] in which a graph presentation to measure centrality is constructed for the entities in a given document. However, the positional baseline as explained in [3] is still a solid baseline. In later work, the recall of salient entities is slightly higher compared to the recall of salient entities of the positional baseline. Examples of recent entity salience detection systems are found in [15], [25] and [19]. In the remainder of this section, an elaboration of these methods is given.

2.2.1 Entity Centrality method

In [3] a binary entity salience estimation task is described. It is one of the first papers which uses a large dataset and the test set of the dataset is used by other works on entity salience as a baseline. The paper describes how a large annotated dataset is created, based on millions of documents from The New York Times. The method proposed a small set of hand-crafted features and then runs Page Rank on top of the entity-entity graph to compute the importance of entities in a given document. Their evaluation shows that entity centrality does not add as much information after the mention features are given.

There is some computational complexity involved in the preprocessing of a document. Named Entity Recognition (NER) is done and also Coreference Resolution is used in order

to find the mentions of a given entity. In order to compute these features on millions of documents, large amounts of computational resources are required.

2.2.2 SEL algorithm

The SEL algorithm as described in [19] consists of two steps. The first step is the Candidate Pruning step which consists of spotting possible mentions of entities. Then, the found candidates are pruned and eventually linked to Wikipedia entities. The second step is the salient linking step. In this step, the remaining candidates are linked to Wikipedia pages and the salience score is estimated. Just like [3], the entity salience problem is formulated as a binary classification task where an entity is either relevant (salient) or irrelevant.

This work describes features related to entity salience and also puts the computational complexity of the features in two classes: "heavy" features and "light" features. This is an interesting separation since not all entity salience research can be compared fairly because methods with high F_1 scores might use many features which require a large number of computational resources. Recent work in sequence-to-sequence models [20] also report the computational complexity in comparison to other models.

One of the missing features is in-document entity-entity relations. This feature is probably an important feature since it allows for connections between Wikipedia entities and non-Wikipedia entities. Not all entities are described on Wikipedia. And even if two entities are described on Wikipedia, it might miss information about two entities which is described in a particular document. Furthermore, the dataset lacks the annotation of phrases and keyphrases and the most important sentences are estimated by the algorithm. However, subsentence information would be interesting to investigate. An extension to the created dataset of this research is described in the dataset section.

2.2.3 Kernel Entity Estimation Model (KESM)

The Kernel Entity Estimation Model (KESM) as described in [25] generates Knowledge Enriched Embeddings (KEE) for each entity. The Knowledge Enriched Embeddings are multi-word representations which incorporate information about an entity in a single vector - similar to word embeddings. These vectors describe each entity by their descriptions. These entity descriptions are fetched from Freebase - which is a Knowledge Base.

The described Kernel Interaction Model (KIM) is an elegant way to estimate entity salience. This model is used for entity-entity interactions. Other works on entity salience often make use of complex preprocessing. This method does not need any complex preprocessing. The features are simple and that makes that this work is easy to reproduce.

The entity descriptions are fetched from Freebase. However, note that entities which are not in the knowledge base have a sparse representation. One counter-argument could be that the authors did not use entities which are not in Freebase and that therefore all entities have a description. However, it is not realistic to assume that all knowledge bases have a complete representation of all entities in the world. In other words, there exists a fair amount of entities without Freebase description. Therefore, these entities have sparse KEE vectors. One other issue with this model is that it does not incorporate the importance of words. Some words (for example words in keyphrases) are more important than other words. However, the model assumes that all words are equally important.

2.2.4 SWAT system

The SWAT system as defined in [15] reports a high recall for their method compared to the other methods. Their system is composed of several subsystems. It generates feature vectors per entity and these vectors consist of ranked sentences per entity, syntactic and semantic relatedness between entities and dependency relations per entity.

One of the differences between this method and the other methods is that this method relies on heavy NLP annotations inside each document. Because the method does lead to a high recall, it might be hypothesized that inter-document relations between entities are an important factor determining salience. It also reports which of the used features are the most important for entity salience. Besides that, it also points out that the rule-based approach used by [3] might be improved by using a crowdsourced ground-truth instead of the rule-based approach. They also point out that the preprocessing of the NYT (New York Times) dataset took about 20 days and suggested to research the design of faster entity linkers.

The system consists of several complex subsystems, which makes it hard to reproduce and these subsystems might even change over time. So, reproducibility is an issue here. One other issue is the number of computational resources needed to compute entity salience. It relies heavily on computationally demanding preprocessing. One other issue arises for entities which are not available in an external knowledge base and another issue is the scalability of this method. There is no expression given for the computation time of the method. From the graphs, it might be assumed that the computation time might increase exponentially if the number of tokens, mentions or entities increases linearly.

2.2.5 Comparing entity salience methods

One of the issues with recent entity salience methods is the issue of entities that are not available in an external knowledge base. If an entity is not linked to an entity in a knowledge base, no entity vector is built. Then, no relation is learned by entities not in the knowledge base and other entities.

Another issue is the computation time. Some methods do work but require heavy preprocessing which might imply an exponentially increasing computation time concerning the number of tokens or the number of entities. These methods might not be well-suited for large text documents.

2.3 Text summarization

More is known on text summarization and the aboutness of text. An overview of extractive text summarization methods is found in [5]. One of the first papers on text summarization using computers is the paper by Luhn [10]. The work on local coherence [4] and the work on discourse trees [11] form the inspiration for the construction of the Wikiphrase dataset. Text summarization is closely related to entity salience detection. The most prominent entities are often found in summaries of a document. Therefore, the paper by Dunietz [3] proposes an automated system for extracting salient entities from texts using the abstracts of the texts.

2.4 Keyphrase extraction

Keyphrase extraction is a form of extractive text summarization in which the most important phrases are extracted from a text. Phrase extraction (and keyphrase extraction) might be important for entity salience detection which is discussed in this thesis. Due to the internet, a large dataset is available for keyphrase extraction since every webpage has a description which often is related to the most important phrases of the webpage itself. Work on keyphrase extraction is found in [6]. The survey highlights several works on keyphrase extraction. For example, in conversational texts, it is known that the topic of the conversation is likely to change over time. It is also known that keyphrases within a document are related to each other. Often, a keyphrase extraction method consists

of two steps: extracting a list of words or phrases as candidate keyphrases and then a classification task to determine whether a candidate keyphrase is a keyphrase or not. However, as noted by [6], binary classification does not distinguish different keyphrases from each other. Commonly used features are within-collection features computed on the training documents (for example TF-IDF features), structural features (e.g. the position of keyphrases within a document), syntactic features (for example the usage of Part-of-Speech tags or morphological suffixes of words) and external resource-based features (e.g. features from knowledge bases such as Wikipedia). The external resource-based features are exploited for determining the salience of keyphrases in [9] which resulted in low precision and recall scores. These results are similar to results on the entity salience task. The authors also mention the attempts that lowered the precision and recall.

2.5 Neural networks

In this section, the building blocks for some of the models created in this thesis are described.

2.5.1 From nervous activity to differentiable systems

The first work on artificial neural networks is the work "A logical calculus of the ideas immanent in nervous activity" [12] in which the link is made between nervous activity and logical calculus. One well-known minimalistic model that is still used to date and forms the basic of most modern neural network models is the perceptron [17]. After the perceptron, backpropagation was introduced [23] and from that time on, end-to-end differentiable models without feature engineering could be build.

2.5.2 Recurrent Neural Networks

Recurrent Neural Networks are Neural Networks that are capable of processing sequences instead of processing only a fixed-size item. After the introduction of the backpropagation algorithm, it was still not possible to create models over sequences in which parameters are shared accross the sequence. An improved version of the backpropagation algorithm was made such that it could handle sequences [24]. One of the open issues of recurrent neural networks was the vanishing gradient problem and one of the first models capable of processing sequences is the LSTM [7]. The LSTM is one of the models used in this thesis.

2.6 Related datasets

In this section, several related datasets are discussed.

2.6.1 SNEIT dataset

The SNEIT dataset (Salient Named Entity Identification in Tweets) consists of a large number of Tweets (short texts, normally consisting of one or a few sentences) with corresponding images and salient entities found in these Tweets. This dataset is described in [16]. In this dataset, an entity is marked as salient for a Tweet if it appears in the image which belongs to the Tweet. An entity is either marked as salient or as non-salient. This dataset is useful for validating the generalizability of entity salience detection methods since it is structurally different from the frequently used news corpora. However, the dataset is not used in this work, since not all salience clues are found in the texts. Some salience clues are found in the images and images fall outside the scope of this work.

Chapter 3

Datasets

This chapter describes entity salience datasets and it describes possible improvements for these datasets. In the remainder of this chapter, it is explained how one of the datasets is enriched with additional information and where the publicly available enriched dataset is found. The process of enriching the dataset is also described such that the results can be replicated and more entity salience datasets can be enriched using a similar method.

3.1 Reproducibility and open datasets

Many different types of documents are used in entity salience research and salience research in general. A large proportion of the documents are news articles. A problem with the documents (and in particular with news articles) is that most of the documents are protected by copyright. If that is the case, it is not easy to reproduce and enhance the results. To encourage progress in science, it is important to choose and use open datasets such that the reproduction of the results is easier. In this work, the Wikinews dataset as described and created in [19] is used and extra annotations are added. This dataset has no copyright constraints ([19]). However, the quality of the articles in this dataset might not be as good as the quality of the articles in the NYT dataset ([3]), which is created by professional journalists. Besides that, it is open source (CC BY 2.5) and the salience annotations are crowd-sourced. Therefore, this is a good starting point for reproducibility.

3.2 Overview of entity salience datasets

Several datasets are described in research on entity salience. In this section, the several entity salience datasets are described.

The Wikinews dataset is described in [19]. It consists of 365 documents and the annotators are asked to mark an entity as either top relevant, highly relevant, partially relevant and not relevant (either when an entity is not relevant or not mentioned at all). Therefore, there are four different discrete entity salience levels found in this dataset.

In that way, the score becomes a score in the interval $[0, 1]$ where 1 equals highly relevant and 0 equals irrelevant. An entity in these datasets is salient if and only if $n(\mathbf{s}) \geq 0.6$ where 0.6 is chosen empirically.

The annotated NYT corpus is described in [3] and it includes annotations for 1.8 million articles published between January 1987 and June 2007 from the New York Times. It has binary entity salience scores since an entity is either marked as salient or non-salient.

In this project, only the test set of the annotated New York Times corpus is used. It is computationally demanding to process the train set of this corpus. However, the test set is important in order to be able to compare newly developed entity salience detection methods with existing methods.

In the following table, a summary is given of the used datasets.

Dataset name	Salience scores	Domain	Document count
Wikinews	Discrete score	News articles	365
New York Times (test set)	Binary	News articles	9706

Table 3.1: A summary of different entity salience datasets.

In the next table, a few statistics are given on the used datasets.

Dataset name	Words per document	Entities per document
Wikinews	257 (±73)	11 (±3)
New York Times (test set)	829 (±440)	19 (±11)

Table 3.2: Statistics of different entity salience dataset properties describing average values and one standard deviation of the counts.

3.2.1 Different document types

There are structural differences in the documents used in several datasets. For example, the NYT dataset ([3]) contains news articles just like the Wikinews dataset ([19]) but the documents of the Wikinews dataset have fewer words than the documents of the NYT dataset. There are also thematic differences. The SNEIT dataset [16] consists of Tweets which mainly consists of documents which contain personal opinions of approximately 140 characters. In these cases, the theme of the Tweets and the theme of news articles differ. In this work, a dataset is used that only contains news articles. The reason for this is that this dataset is publicly available and the results are easy to reproduce. In the future, it might be interesting to enlarge the domain of the documents in the dataset such that scalability of the methods can be evaluated to answer questions like: "How well does a method work on short documents and long documents?". It would be interesting to see which features only work for news documents and which features work for Tweets. However, that question falls outside the scope of this research.

3.3 A new entity salience dataset

Much research is done on keyphrase extraction. In the task of keyphrase extraction, the most important phrases are extracted from a text. However, it is not the case that a keyphrase only consists of salient entities. Often, a keyphrase will contain a mix of salient and non-salient entities. It is interesting to zoom in more on the relation between (key)phrases and salient or non-salient entities. One goal of the new annotations is to augment an existing entity salience dataset with (key)phrases which contain at least one entity.

3.4 The Wikiphrase dataset

In this work, the Wikiphrase dataset is created. As pointed out by [2], there is a need for more human-annotated training corpora. This could improve algorithm adaptation and parameter tuning.

In order to overcome most of the issues with the current datasets, we set out to create a new dataset satisfying the following desiderata:

- The preprocessing of the dataset should not rely on commonly used NLP tools for extracting entities and for extraction information depending on entities.
- It should be easy to reproduce the results and there should not be a copyright constraint on the newly created dataset such that this dataset can be made publicly available.

In order to overcome the first problem, an abstract concept of a "phrase" is introduced (different from other notions of phrases). Note however that the preprocessing then depends on the choices of the person who is responsible for annotating the text; but, it is useful for the comparative evaluation of methods independent from the used preprocessing pipelines. In the remainder of this work, it is researched whether there exists a relation between phrases and entity salience.

By only using and enhancing datasets without copyright constraints, results based on these datasets are publicly available and easy to reproduce.

3.5 Commonly used Natural Language Processing tools

Some entity salience classification methods (e.g. [3] and [19]) rely on the same Natural Language Processing pipelines (NLP). In order to reduce the effort of the specific NLP preprocessing pipeline, a dataset is created in which phrases are selected for each entity in which a phrase might describe relations between entities and attributes of entities (Wikiphrase). The phrases serve as a unit of abstraction of the NLP tools and might shed some light between the relation of the choice of the NLP tools and entity salience. One other goal of the new dataset is to evaluate commonly used NLP tools using the concept of phrases.

3.6 Creating the dataset

In this project, the Wikinews entity salience dataset as described in [19] is augmented with (key)phrase information. The task description and annotation process are explained in this section.

The dataset is publicly available on the code repository of this project. It was created by two annotators and it is an enrichment of the Wikinews dataset. Not all of the document-entity pairs were annotated, but only a subset; due to time constraints, only the first 100 articles of Wikinews were annotated. The analysis of this dataset is found in the results chapter.

3.6.1 Task description

The Wikinews dataset as described in [19] on which the enrichments are based consists of documents D and entities E. Furthermore, document-entity relations are given (D, E) and an entity salience score e is given for each (D, E) pair. So the entity salience dataset consists of tuples (D, E, e). This dataset is augmented with phrases. Here, a phrase is defined as a subsequence of words in a document that is about a given entity. As a consequence, it can describe relations between entities and attributes about one or more entities. Here, phrases contain 15 words on average. In this task, each document-entity pair is augmented with a set of phrases, such that the dataset consists of tuples (D, E, P) where P represents a phrase. Each phrase is either a keyphrase $(p = 1)$ or a non-keyphrase $(p = 0)$ and this phrase salience score is denoted by p. Therefore, the original Wikinews dataset consists of tuples (D, E, e) and is augmented with tuples (D, E, P, p).

3.6.2 The annotation process

For the phrase annotations, BRAT [18], a web-based tool for text annotations is used.

The following figure shows BRAT being used during the annotation process.

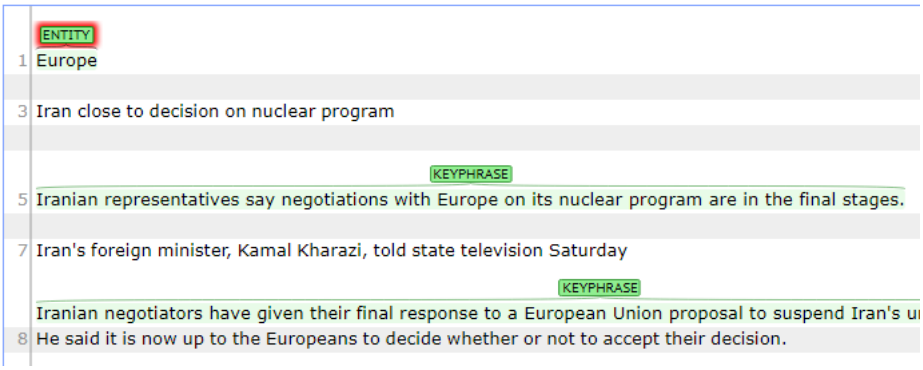

Figure 3.1: A screenshot of the phrase annotations in which two keyphrases are annotated for one given document-entity pair (D, E).

For each document-entity pair, the entity is shown on the first line to the annotator. The following line displays the title of the document and the remaining lines contain the contents of the document. The annotator is asked to select all phrases in either the document contents or in the title which contain the given entity or a mention to the given entity. When a phrase is selected, the annotator can choose to mark a phrase as keyphrase.

Chapter 4

Representation

4.1 Entity salience detection based on phrases

In this section is explained why and how entity phrases are used to estimate entity salience.

4.2 Why phrases for entity salience detection?

Here, a phrase is defined as a subsequence of words in a document that is about a given entity and describes one relation between one or more entities. The number of phrases relates to the number of entity mentions, the position of phrases is an indicator for the position of entity mentions and the collection of the phrases for a particular entity summarizes the contribution of the given entity in a document. Therefore, phrase information is useful for entity salience estimation as shown in the chapter on the results.

In previous research, Natural Language Processing (NLP) techniques are used to estimate entity salience. By using algorithms, some errors are introduced and therefore important information about an entity within one document might be lost which might decrease the recall score for the entity salience task. Notice that the recall score for salient entities for the positional baseline is low. In order to improve upon the positional baseline, it might be useful to decrease the errors made by NLP techniques such that the recall score increases.

First of all, a phrase is a unit of abstraction for commonly used NLP subtasks, such as Named Entity Recognition and Coreference Resolution. It is a reduction in the amount of work for an annotator since the annotator does not have to label entities and label mentions and draw connections between the mentions and the entities.

Besides that, not only mentions are needed for entity salience detection, but also the information related to the entity or its mention. By using phrases, this type of information is selected and can be used by any entity salience detection algorithm.

However, by only using phrases, the information about the exact positions of mentions of an entity within these phrases is lost. A phrase about a mention of an entity gives an estimate of the position of a mention, but the information about the position of a mention within the phrase is not specified.

4.2.1 Concept of a phrase by example

The following example illustrates the concept of phrases:

Alice and Bob were into cryptography. Alice sents an encrypted message to Bob, but Bob lost all of his keys. Therefore, Bob was unable to read her messages. After a while, Bob managed to restore all of his keys. He was able to decrypt all the messages.

The following phrases could be found in the example:

- Alice and Bob were into cryptography
- Alice sents an encrypted message to Bob
- Bob lost all of his keys
- Bob was unable to read her messages
- Bob managed to restore all of his keys
- He was able to decrypt all the messages

This is a subjective task, so different annotators could come up with different phrases.

4.3 How to use phrases for entity salience detection?

First, phrase vectors are computed which are a numerical representation of the phrases. For each of the phrases, a positional encoding is computed and a semantic encoding is computed. The concatenation of both vectors is called the phrase vector in this work.

4.4 Semantic phrase vectors

The semantic phrase vectors are a combination of the word vectors. Suppose that a phrase consists of k consecutive words $w_1, ..., w_k$. For most of the words, there is a corresponding word vector. Let $E(x)$ be an embedding function, that takes a word x and computes the corresponding word vector. Then, for each word w_i we can compute a word vector v_i:

$$v_i = E(w_i) \tag{4.1}$$

Then, the phrase vector p, consisting of the words $w_1, ..., w_k$ and corresponding word vectors $v_1, ..., v_k$, is computed by taking the average of the word vectors:

$$p = \frac{1}{k} \sum_{i=1}^{k} v_i \tag{4.2}$$

4.5 Positional encodings for phrases

As pointed out by [3], entity salience is mainly based on the positions of entities in a text. Positional encodings or positional embeddings are used in [20]. Commonly used Recurrent Neural Network architectures are computationally demanding since a state vector is computed by processing the words in a text one-by-one. It is hard to parallelize such architecture because of this dependency. To get rid of the dependency, the authors of [20] used positional embeddings. These vectors are a numerical representation of a position. Then, a feed-forward neural network can be trained (without recurrence) using the positional information and semantic information of words.

In this work, the positional encodings of phrases are combined with the semantic information of the phrases. Thus, for each phrase, one vector is computed such that it is a concatenation of the combined phrase vector containing semantic information of the phrase and a positional encoding for the position of the phrase in the document.

In the earlier work on positional embeddings [20], the positional embeddings are computed as follows:

$$PE(p, 2d) = \sin\left(\frac{p}{M^{\frac{2d}{D}}}\right) \tag{4.3}$$

$$PE(p, 2d+1) = \cos\left(\frac{p}{M^{\frac{2d}{D}}}\right) \tag{4.4}$$

Where p is the position, d is the dimension, D is the number of model parameters and M is a constant. In the paper, M is fixed to 10000. In this work, M is set to approximately the maximum number of words in a document.

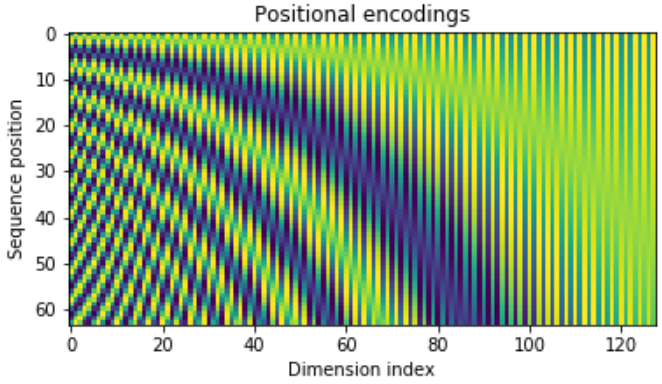

Figure 4.1: Positional embeddings with 128 dimensions, 64 positions and $M = 32$.

As can be seen from figure 4.1, vectors with a similar position have more similar positional vector representations. The positional information is encoded in the dimensions but the embedding cannot encode more positions than dimensions. Therefore, it must roughly hold that $D \geq P$ where D is the number of dimensions and P is the maximum number of positions.

The positional encodings have a powerful property. For any fixed offset k, the positional encoding $PE(x+k, 2d)$ can be represented as a linear function of $PE(x, 2d)$ and $PE(x, 2d+1)$ (a similar argument is found for $PE(x+k, 2d+1)$). Let $f(d) = M^{\frac{2d}{D}}$. Note that:

$$\begin{aligned}
PE(x+k, 2d) &= \sin(\frac{x+k}{f(d)}) && (4.5)\\
&= \sin(\frac{x}{f(d)} + \frac{k}{f(d)}) && (4.6)\\
&= \sin(\frac{x}{f(d)})\cos(\frac{k}{f(d)}) + \cos(\frac{x}{f(d)})\sin(\frac{k}{f(d)}) && (4.7)\\
&= PE(x, 2d)PE(k, 2d+1) + PE(x, 2d+1)PE(k, 2d) && (4.8)\\
&= \begin{pmatrix} PE(x, 2d) \\ PE(x, 2d+1) \end{pmatrix} \cdot \begin{pmatrix} PE(k, 2d+1) & PE(k, 2d) \end{pmatrix} && (4.9)\\
&= PE_x \cdot PE_k^T && (4.10)
\end{aligned}$$

Where $\cdot$ is used as the dot product and where $PE_x = \begin{pmatrix} PE(x, 2d) \\ PE(x, 2d+1) \end{pmatrix}$ and $PE_k = \begin{pmatrix} PE(k, 2d+1) \\ PE(k, 2d) \end{pmatrix}$.

Since the positional encodings are linear, the paper [20] hypothesized that this would allow that the model could learn to attend a relative offset. In general, it can be hypothesized that any linear function of positional encodings might learn to attend to a relative offset.

In previous work on entity salience, hand-crafted positional features are used. For example, one can encode the position of a word in a text by a number of bags. The first bags representing the first number of words, the second bag representing the number of words after the word positions represented by the first bag and in general, the n^{th} bag representing the numbers of words after the word positions represented by the $(n-1)^{\text{th}}$ bag. The advantage of positional encodings are the relative positional offsets that can be learned. In this work, positional embeddings are used.

4.6 Entity vectors

One entity belongs to one or more phrases. After computing the semantic phrase vectors and positional encodings of the phrases belonging to the entity, the aggregated phrase vector is computed. Phrase vector m_i is a concatenation of the semantic phrase vector s_i and the positional embedding p_i:

$$m_i = [s_i; p_i] \tag{4.11}$$

Then, all phrase vectors belonging to entity E are combined by computing the average of all these phrase vectors:

$$z_i = \frac{1}{|P(E)|} \sum_{p \in P(E)} p \tag{4.12}$$

Where z_i is the entity vector of the i^{th} entity, $P(E)$ is the set of phrase vectors belonging to entity E and $|P(E)|$ is the number of phrase vectors belonging to entity E.

By computing z_i, a fixed-size entity vector is computed for each entity.

4.7 Document vectors

The document vector is computed by averaging all phrase vectors. Let d be the document vector for a given document. Let E be the collection of all entities in the given document and $|E|$ be the number of entities in the given document. Then:

$$d = \frac{1}{|E|} \sum_{i=1}^{|E|} z_i \tag{4.13}$$

So the document vector is a linear combination of all the entity vectors.

Chapter 5

PhraseNLP: Phrase extraction using NLP preprocessing

5.1 Method description

The phrases as defined in the dataset chapter contain at least one entity or entity mention. So by finding all entities (using Named Entity Recognition) and by finding all of its mentions (using Coreference Resolution), one can extract sentences that contain at least one entity or entity mention. Notice that the PhraseNLP method depends on a NLP toolkit. The method of finding all sentences with contain at least one mention to a given entity is called PhraseNLP in this work. Phrases are sequences of words within sentences, but this subsentence information is lost since only complete sentences are used. Then, the entity salience classifier is used on all extracted sentences to estimate the entity salience. The PhraseNLP method is also used to evaluate NLP toolkits on the Wikiphrase dataset.

5.2 Comparing Wikiphrase and PhraseNLP

What are weaknesses in current NLP systems and what are weaknesses in the manual phrase annotations given by Wikiphrase? This question is answered by comparing the Wikiphrase dataset with PhraseNLP annotations. In this work, phrases are sequences of words within one sentence expressing a relation between one or more entities or entity mentions. Therefore, a phrase should contain at least an entity or a mention (obtained by Coreference Resolution). Here, we deploy the NER and coreference resolution components of Stanford NLP (Stanford CoreNLP 3.9.1). The manually annotated phrases from Wikiphrase are compared to the NER and Coreference Resolution results from the Stanford CoreNLP toolkit. The following questions are researched. What mentions are contained in phrases but not found by the NLP toolkit? And what mentions are found by the NLP toolkit but not contained in a phrase? This might reveal weaknesses both in the manual annotations and in the NLP toolkit used by the PhraseNLP method.

5.3 An example

Consider the following text:

Alice and Bob were into cryptography. Alice sents an encrypted message to Bob, but Bob lost all of his keys. Therefore, Bob was unable to read her messages. After a while, Bob managed to restore all of his keys. He was able to decrypt all the messages.

Using PhraseNLP method, all sentences containing at least one reference to a given entity are extracted. Suppose that the given entity is "Alice". The first step is to extract

all of its mentions using Named Entity Recognition and Coreference Resolution:

Alice and Bob were into cryptography. **Alice** sents an encrypted message to Bob, but Bob lost all of his keys. Therefore, Bob was unable to read **her** messages. After a while, Bob managed to restore all of his keys. He was able to decrypt all the messages.

Then, the entity sentences (sentences with at least one mention about the entity "Alice") extracted by the PhraseNLP method are the following:

1. **Alice** and Bob were into cryptography.
2. **Alice** sents an encrypted message to Bob, but Bob lost all of his keys.
3. Therefore, Bob was unable to read **her** messages.

Chapter 6

PhraseSeq2Seq: Phrase extraction using a sequence-to-sequence model

6.1 Method description

The phrase extraction model should give an answer to the following question: given a document-entity pair and given a fragment of the document, which words in the fragment belong to a phrase about the given entity? In this chapter, the PhraseSeq2Seq model is described which answers the question using a sequence-to-sequence model.

The Wikiphrase dataset contains phrase annotations for document-entity pairs. Therefore, this dataset is useful for answering the question.

To know what phrases belong to a given entity, the entity and additional information about the entity should be encoded such that it can be used as input. After processing a fragment, the entity encoding should be updated such that new information found in the fragment about the entity is added to this encoding.

A fragment (or entity or phrase or document) consists of words and words consist of characters. Most character-based models processing documents consisting of many words are slow on moderate computing resources. Therefore, the phrase extraction model described here will be word-based. That is, the words are the smallest units of information found in texts. Pretrained word embeddings are used in order to speed up the learning process and these numerical representations of words are then used as inputs for the model. Also, the entity information is encoded the same way as the fragment is encoded.

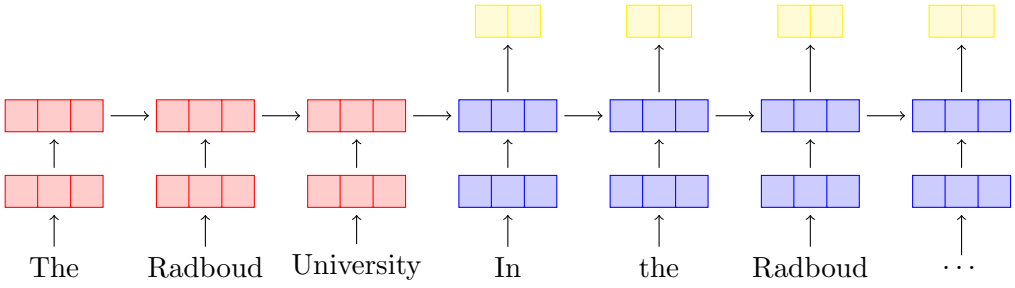

Figure 6.1: An overview of a sequence-to-sequence model. Left (in red) is the encoder for encoding a fragment and on the right (in blue) is the decoder which can produce tokens. Here, the encoder encodes an entity ("The Radboud University") and it marks tokens belonging to phrases about the Radboud University using the decoder.

In figure 6.1, a schematic view of a sequence-to-sequence model is given. A sequence-to-sequence model encodes information and then decodes the encoded information by producing the output token by token.

Simpler architectures are also possible. For example, it would be possible to use just an encoder. This is equivalent by replacing the decoder with the encoder.

The end-to-end process is as follows. First, the entity (and extra information from a knowledge base) is converted to words and the numerical representations of the words are used as input for the encoder (for which the initial hidden state is set to zero). The encoder then generates a hidden state by updating its state word for word. The decoder then uses the final hidden state of the encoder as its initial hidden state. Words are extracted from the fragment and these words are converted to their numerical counterparts. Then, the decoder updates its hidden state word for word. A linear transformation is computed on the intermediate hidden states and a softmax transformation is computed on top of it such that a probability distribution of two classes is computed. Then, binary cross entropy is used as a loss function and the error is backpropagated through the network and all weights are updated. The outputs from the softmax transformation are a probability distribution over two classes; the given word is either in a phrase or not in a phrase. The ground truth of whether a word belongs to a phrase for a given entity pair is found in the Wikiphrase dataset.

6.1.1 Text representation

The words need first to be converted into numerical representations. However, learning numerical representations for words is a time-consuming task for which a lot of computational resources are needed. Pretrained word vectors are available which are trained on millions of documents. One example of pretrained word vectors are GloVe vectors [14]. The lowercase variants of the words from which all accents are removed are looked up in the GloVe vocabulary. Then, the corresponding numerical representations are used by the model.

6.1.2 Output representation

A softmax classifier is trained on the hidden states of the decoder. The outputs are a probability distribution over two classes; a word is either in a phrase or not in a phrase.

6.1.3 Loss function

Binary cross entropy is used as a loss function. This is due to the fact that the output is a probability distribution over a binary variable per word. Suppose there are n words to predict the phrases for. Given a vector of targets $t = \begin{pmatrix} t_1 & t_2 & \dots & t_n \end{pmatrix}^T$ with $t_i \in \{0, 1\}$ where $t_i = 1$ when the i^{th} word belongs to a phrase and $t_i = 0$ otherwise and given a vector of predictions $y = \begin{pmatrix} y_1 & y_2 & \dots & y_n \end{pmatrix}^T$ where y_i corresponds to the i^{th} word and satisfies the probability axioms. Binary cross entropy loss is defined as the following:

$$L = \frac{1}{n} \sum_{i=1}^{n} t_i \cdot \log(y_i) + (1 - t_i) \cdot \log(1 - y_i) \tag{6.1}$$

The loss function is then optimized (here: minimized) by one of the optimizer algorithms as described in the related work chapter.

6.1.4 Memory cells

Neural networks in which a state is depending on previous states are called Recurrent Neural Networks (RNNs). There are different types of RNNs. The GRU (Gated Recurrent Unit) [1] is a frequently used memory cell. GRUs are faster than Long Term Short Term Memory cells (LSTMs) but functionally approximately equivalent. Therefore, the GRU is used in the sequence-to-sequence model.

Chapter 7

Entity salience classification methods

In this chapter, the created entity salience classification methods are described.

7.1 Evaluation of entity salience on Wikiphrase

In order to evaluate how phrases contribute to entity salience, the precision, recall and F_1 scores are computed for both salient and non-salient entities. In the following sections is explained how the phrases are used to estimate entity salience.

7.2 Entity salience classification

First, a document-entity vector is computed for each document-entity pair. This vector is a concatenation of the document vector and the entity vector and the difference between the document vector and entity vector for each of the pairs. Then, one of the entity salience classifiers is used to determine the entity salience. The used entity salience classifiers are described in this section.

7.2.1 ESNN: Entity Salience Neural Network classifier

The ESNN is a two-layered neural network is used with ReLU activations and 64 hidden units per layer used for entity salience classification. The output is one number and the target is the binary entity salience score which is 1 if the entity is salient and 0 otherwise.

7.2.2 ESRF: Entity Salience Random Forest classifier

The ESRF is a Random Forest classifier used for entity salience classification. Here, 10 estimators are used and Gini impurity is used as the value the optimize for. The minimum sample split is set to 2.

Chapter 8

mESD: Entity Salience Detection and Phrase Extraction method

In this chapter the mESD (mLSTM Entity Salience Detection) model is introduced for jointly learning entity salience detection and entity phrase extraction.

8.1 Training procedure

For the training, both the Wikiphrase dataset and the NYT dataset are used. The Wikiphrase dataset is manually annotated and contains less than 1000 training samples. On the other hand, the NYT dataset contains millions of training samples. However, the NYT dataset does not contain phrase annotations. In order to train for both tasks, a random sample (consisting of multiple document-entity pairs) is picked from the Wikiphrase dataset and a random sample (consisting of multiple document-entity pairs) is picked from the NYT dataset. Then, the losses are computed. For the Wikiphrase dataset, the phrase loss is computed and for the NYT dataset the salience loss is computed. Both losses are averaged and the averaged loss is minimized.

8.2 Loss functions

8.2.1 Phrase loss

Each word in a document either belongs to an entity phrase or not. The phrase loss ($L^{(phrase)}$) is the binary cross entropy loss computed for the predicted labels (either inside a phrase or outside a phrase) and the ground-truth labels.

8.2.2 Salience loss

For each document-entity pair in a batch, a salience ground-truth label is given. The model predicts the saliency class for each document-entity pair and binary cross entropy loss is computed ($L^{(salience)}$) for the predicted labels and the ground-truth labels.

8.2.3 Combined loss

In this method, two tasks are learned simultaniously. The task of phrase extraction is learned and the task of entity salience classification is learned. Let $L^{(phrase)}$ be the phrase extraction loss and $L^{(salience)}$ be the salience classification loss. Then, the global loss is computed as follows:

$$L = \frac{1}{2}L^{(phrase)} + \frac{1}{2}L^{(salience)} \tag{8.1}$$

The optimization algorithm then minimizes this global loss function L.

Precision, recall and F_1 score for random binary classification

What is the precision, recall and F_1 score for random binary classification? Let x be the prediction of the model and assume that the model predicts randomly one of $\{0, 1\}$. Then $P(x = 0) = \frac{1}{2}$ and $P(x = 1) = \frac{1}{2}$. Let y be the ground-truth binary label and assume that α is the ratio of ground-truth labels equal to 1 (so $1 - \alpha$ is the ratio of ground-truth labels equal to 0). Assume that the labeling of the ground-truth items does not depend on the random selection procedure for x. Then, the following holds:

$$P(x = 1, y = 1) = P(y = 1|x = 1)P(x = 1) = P(y = 1)P(x = 1) = \frac{1}{2}\alpha \tag{8.2}$$

$$P(x = 1, y = 0) = P(y = 0|x = 1)P(x = 1) = P(y = 0)P(x = 1) = \frac{1}{2}(1 - \alpha) \tag{8.3}$$

$$P(x = 0, y = 1) = P(y = 1|x = 0)P(x = 0) = P(y = 1)P(x = 0) = \frac{1}{2}\alpha \tag{8.4}$$

$$P(x = 0, y = 0) = P(y = 0|x = 0)P(x = 0) = P(y = 0)P(x = 0) = \frac{1}{2}(1 - \alpha) \tag{8.5}$$

Suppose that m items are labeled. Then, it is possible to compute the expectations of the true positive count (TP), false positive count (FP), false negative count (FN) and the true negative count (TN) by the following equations:

$$TP = P(x = 1, y = 1)m = \frac{1}{2}\alpha m \tag{8.6}$$

$$FP = P(x = 1, y = 0)m = \frac{1}{2}(1 - \alpha)m \tag{8.7}$$

$$FN = P(x = 0, y = 1)m = \frac{1}{2}\alpha m \tag{8.8}$$

$$TN = P(x = 0, y = 0)m = \frac{1}{2}(1 - \alpha)m \tag{8.9}$$

Now, the precision, recall and F_1 score can be computed in terms of α. For the precision (P) it holds that:

$$P := \frac{TP}{TP + FP} = \frac{\frac{1}{2}\alpha m}{\frac{1}{2}\alpha m + \frac{1}{2}(1 - \alpha)m} = \alpha \tag{8.10}$$

In other words, the expected precision for a task where half of the items belong to the positive class and half of the items belong to the negative class where all predicted labels are chosen randomly with probability $\frac{1}{2}$ is $\frac{1}{2}$. For a task with only $\frac{1}{10}$ of the labels belonging to the positive class and randomly chosen predicted labels, the expected precision is $\frac{1}{10}$.

For the expected recall (R), the following result is obtained:

$$R := \frac{TP}{TP + FN} = \frac{\frac{1}{2}\alpha m}{\frac{1}{2}\alpha m + \frac{1}{2}\alpha m} = \frac{1}{2} \tag{8.11}$$

The expected recall of randomly labeled items does not depend on the distribution of the imbalanced classes.

For the expected F_1 score, the following quantity can be computed:

$$F_1 := 2 \cdot \frac{P \cdot R}{P + R} = \frac{\alpha}{\alpha + \frac{1}{2}} \tag{8.12}$$

Since $\alpha \in [0, 1]$ for the randomly labeled items, it must hold that $0 \leq F_1 \leq \frac{2}{3}$.

In conclusion, for a binary classification task in which a ratio α of all items is equal to 1 and $1 - \alpha$ of all items are equal to 0 for $\alpha \in [0, 1]$ and in which all items are labeled randomly by a model, the precision is equal to α, the recall is equal to $\frac{1}{2}$ and the F_1 score is equal to $\frac{\alpha}{\alpha + \frac{1}{2}}$.

8.3 mLSTM

This method is inspired by the Match-LSTM (mLSTM) method as described in [22]. Their model consists of both a mLSTM and an Answer Pointer network. The Match-LSTM was first introduced by [21]. The model (consisting of the mLSTM and the Answer Pointer network) is used for answering questions and is trained on the SQuAD (Stanford Question Answering Dataset) dataset. The model learns to find mentions of words in the question and this is closely related to the task of finding entities (and entity mentions) in a document. Therefore, the mLSTM could be useful for the phrase extraction task. In this section, the mLSTM method is reformulated in the context of documents and entities and describes the mLSTM model used for entity salience detection.

8.3.1 Word encodings

In our method, GloVe embeddings are used to represent the words in both the text and the entities. Here, $e^{(doc)}$ represents the word embeddings for the document and $e^{(ent)}$ represents the word embeddings for the entity. Three special tokens are added to the vocabulary. An `PAD` token is added for applying padding and for forming batches. An `UNK` token is added for representing tokens that are Out-Of-Vocabulary (OOV) and a `NULL` token is added which is inserted as last token after the entity tokens. The `NULL` token is used by the Match-LSTM such that it represents none of the tokens of the entity.

8.3.2 Text preprocessing

As described in [22], first two one-directional LSTMs are trained on the data. Here, two one-directionals LSTMs are trained on the word embeddings of the text and on the word embeddings of the entities. Let $H^{(doc)}$ denote all hidden states of the LSTM for evaluating the LSTM over the document word embeddings $e^{(doc)}$ and let $H^{(ent)}$ denote all hidden states of the LSTM for evaluating the LSTM over the entity word embeddings $e^{(ent)}$. $H^{(doc)}$ and $H^{(ent)}$ are computed as follows:

$$H^{(doc)} = \overrightarrow{LSTM}(e^{(doc)}) \tag{8.13}$$

$$H^{(ent)} = \overrightarrow{LSTM}(e^{(ent)}) \tag{8.14}$$

Here, $H^{(doc)}$ is a $\mathbb{R}^{l \times |D|}$ matrix and $H^{(ent)}$ is a $\mathbb{R}^{l \times |E|}$ matrix where $|E|$ is the number of entity word tokens, $|D|$ the number of document word tokens and l is the number of dimensions used.

Let $h_i^{(doc)}$ represent the encoded version of the i^{th} word token of the document (so the i^{th} column vector of $H^{(doc)}$) and let $h_i^{(ent)}$ represent the encoded version of the i^{th} word token of the entity (so the i^{th} column vector of $H^{(ent)}$).

8.3.3 The Match-LSTM

The next layer is the Match-LSTM (mLSTM) layer. It sequentially iterates over all words in the document and computes attention scores per word in the document over the words in the entity. It consists of a forward and a backward component. The forward component initializes a hidden state to all zeros. It then gradually updates it internal states by iterating over the document words. In order to update the internal state, the following quantity is computed (in the forward direction).

$$\overrightarrow{G}_i = \tanh(W^{(ent)}H^{(ent)} + (W^{(doc)}h_i^{(doc)} + W^{(r)}\overrightarrow{h}_{i-1}^{(r)} + b^{(doc)} \otimes e_{(|E|)}) \tag{8.15}$$

Here, $W^{(doc)}, W^{(ent)}, W^{(r)} \in \mathbb{R}^{l \times l}$ and $b^{(doc)} \in \mathbb{R}^{l \times 1}$ where $e_{(|E|)}$ copies vector $b^{(doc)}$ $|E|$ times.

$\overrightarrow{G}_i$ relates the entity words with the document words and is used to compute attention over the entity words. The attention is computed as follows:

$$\overrightarrow{\alpha}_i = \text{softmax}(w^T\overrightarrow{G}_i + b \otimes e_{(|E|)}) \tag{8.16}$$

Where $w \in \mathbb{R}^{l \times 1}$ and $b \in \mathbb{R}$. As a result, the $\overrightarrow{\alpha}_{i,j}$ value represents the amount of matching between the i^{th} token in the document and the j^{th} token in the entity.

Then, $\overrightarrow{\alpha}_i$ is used for computing a weighted vector for the entity and combines it with the current document vector:

$$\overrightarrow{z}_i = \begin{bmatrix} h_i^{(doc)} \\ H^{(ent)}\overrightarrow{\alpha}_i^T \end{bmatrix} \tag{8.17}$$

And then, this vector is used as input for a one-directional LSTM:

$$\overrightarrow{h}_i^{(r)} = \overrightarrow{LSTM}(\overrightarrow{z}_i, \overrightarrow{h}_{i-1}^{(r)}) \tag{8.18}$$

All forward quantities are also computed in the reverse direction.

And finally, all hidden states are gathered (in both the forward direction and backward direction):

$$\overrightarrow{H}^{(r)} = \begin{bmatrix} \overrightarrow{h}_1^{(r)} & \overrightarrow{h}_2^{(r)} & \dots & \overrightarrow{h}_{|D|}^{(r)} \end{bmatrix} \tag{8.19}$$

$$\overleftarrow{H}^{(r)} = \begin{bmatrix} \overleftarrow{h}_1^{(r)} & \overleftarrow{h}_2^{(r)} & \dots & \overleftarrow{h}_{|D|}^{(r)} \end{bmatrix} \tag{8.20}$$

$$\tag{8.21}$$

These two matrices are then combined into one matrix:

$$H^{(r)} = \begin{bmatrix} \overrightarrow{H}^{(r)} \\ \overleftarrow{H}^{(r)} \end{bmatrix} \tag{8.22}$$

Where $H^{(r)} \in \mathbb{R}^{2l \times |D|}$.

8.4 The Phrase Classifier

A softmax layer is applied on top of the computed $H^{(r)}$ quantities:

$$y_{in-phrase} = \text{softmax}(W^{(s)} H^{(r)}) \tag{8.23}$$

Where $W^{(s)} \in \mathbb{R}^{2l \times 2}$. This layer is used for computing the phrase loss. When the prediction predicts the positive class, then the document word belongs to a phrase and otherwise the document word does not belong to a phrase.

8.5 The Salience Classifier

A softmax layer is applied on the average of all computed $H^{(r)}$ quantities for computing the salience score:

$$y_{salience} = \text{softmax}(\frac{1}{|D|} \sum_{i=1}^{|D|} H_i^{(r)}) \tag{8.24}$$

This layer is used for computing the salience loss.

8.6 The mESD model

The mESD model is the combination of the mLSTM model and the phrase classifier and salience classifier and the combined loss function is minimized in order to train the model.

Chapter 9

Results

9.1 Positional baseline

The positional baseline is described by most recent research on entity salience. An entity is marked as salient if it appears in the first sentence of a text. Note that this method is not applicable to short texts which consist of only one sentence such as Tweets since it would classify every entity as salient. There are several variants of the positional baseline. For example, one might look at the first n words of a text and test whether an entity appears in these words. Here, the simple positional baseline is computed in which an entity is marked as salient if an entity is mentioned in the first sentence. For determining sentence boundaries, the English Punkt model is used [8]. In the remainder of this section, the positional baseline is evaluated for the dataset described in this work (reported in table 9.1), the Wikinews dataset [19] (reported in table 9.2) and on the annotated test set of the NYT (The New York Times) dataset [3] (reported in table 9.3).

Salience class	**Support**	**Precision**	**Recall**	F_1 **score**
Non-salient entities	999	0.63	0.92	0.75
Salient entities	1378	0.91	0.60	0.72
All entities	2377	0.77	0.76	0.73

Table 9.1: The positional baseline results of Wikiphrase based on the index of the sentence in which the first mention of an entity occurs.

Salience class	**Support**	**Precision**	**Recall**	F_1 **score**
Non-salient entities	3050	0.82	0.94	0.88
Salient entities	1294	0.78	0.53	0.63
All entities	4344	0.80	0.73	0.76

Table 9.2: The Wikinews dataset positional baseline results based on the index of the sentence in which the first mention of an entity occurs.

Salient class	**Support**	**Precision**	**Recall**	F_1 **score**
Non-salient entities	162307	0.91	0.95	0.93
Salient entities	24773	0.56	0.40	0.46
All entities	187080	0.73	0.67	0.70

Table 9.3: The New York Times annotated dataset positional baseline results based on the index of the sentence in which the first mention of an entity occurs.

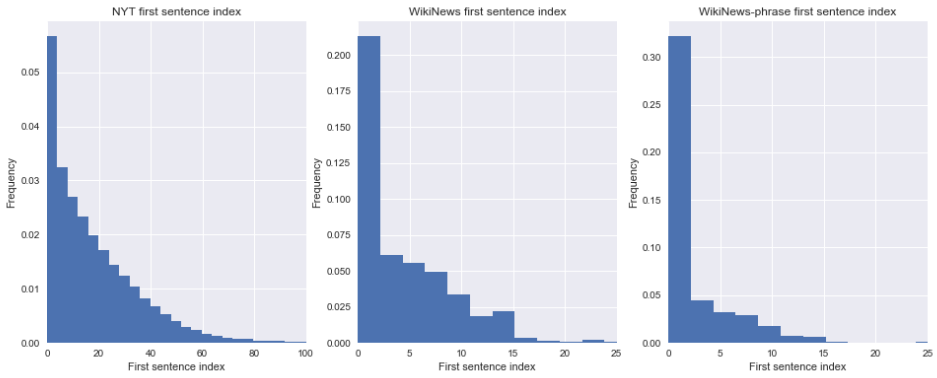

Figure 9.1: Distribution of the first mention sentence index for entities.

The position of the first mention of an entity is a strong indicator of its salience. However, note that the recall of the salient entities is low for this baseline. Figure 9.1 shows the distribution of the first mention sentence index. Most entities are introduced and therefore mentioned in the first few sentences. The figure hints at a long-tail distribution for the first mention sentence index.

Figure 9.2 shows the first mention sentence index for both salient entities and non-salient entities. Here, it can be seen that roughly 50% of all entities mentioned in the first sentence is salient. The proportion of salient entities gets smaller as the sentence index increases.

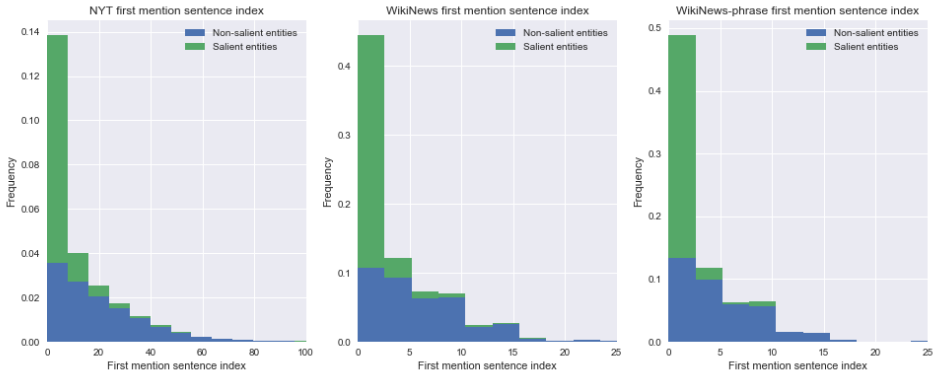

Figure 9.2: Distribution of the first mention sentence index for entities given the salience class.

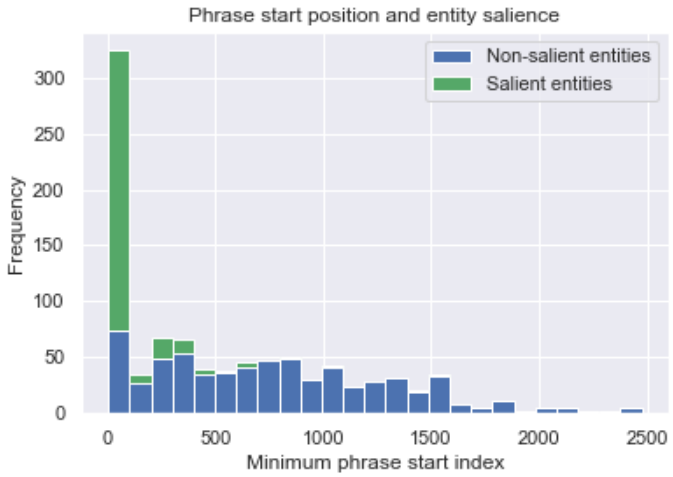

Figure 9.3: Distribution of entity salience given the first entity phrase start index.

9.2 Entity phrases and entity salience

In this section, the results are shown for the relation of the entity phrases in the Wikiphrase dataset and entity salience. The following question is answered: does a relation exist between entity phrases and entity salience?

9.2.1 Phrase start position and entity salience

Here, the phrase start position (the number of character before the first phrase in a document) is compared to entity salience. The results are shown in figure 9.3. It confirms the positional baseline; if the start index is 0, then the probability of an entity being salient is large. However, if the start index is large, then the probability of an entity being salient decreases.

9.2.2 Phrase count and entity salience

It is well known that the entity frequency is an important feature for entity salience. The same question can be asked for phrases: is there a relation between the number of entity related phrases in a document and entity salience? There is indeed a relation between the phrase count and entity salience. The relation is shown in figure 9.4. The more phrases for an entity in a given document, the higher the probability of the entity being salient.

9.2.3 Similarity of mean phrase vector and the document vector and entity salience

Here, the distance between the mean phrase vector (the average of all word vectors of all phrases) and the document vector (the average of all word vectors of the document) is compared to entity salience. Here, distance is defined as the sum of the squared components of the difference between the two vectors. The result is shown in figure 9.5. The smaller the distance, the higher the probability of an entity being salient. Note however that the probability of an entity being salient is roughly smaller than $\frac{1}{2}$.

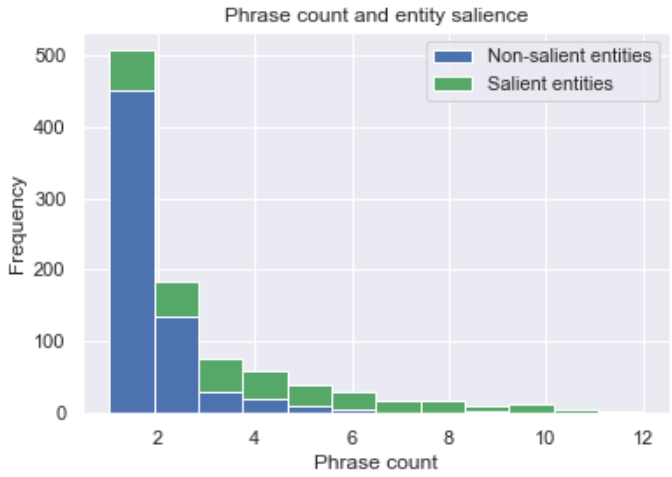

Figure 9.4: Distribution of entity salience given the entity phrase count.

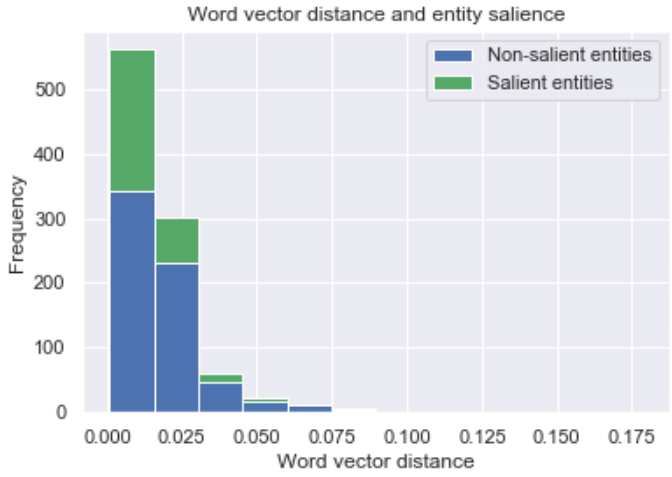

Figure 9.5: Distribution of entity salience given the distance between the document vector and the average phrase vector.

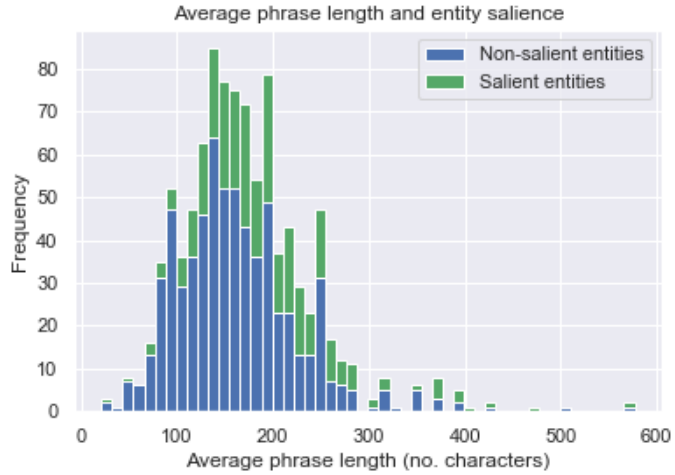

Figure 9.6: Distribution of entity salience given the average phrase length.

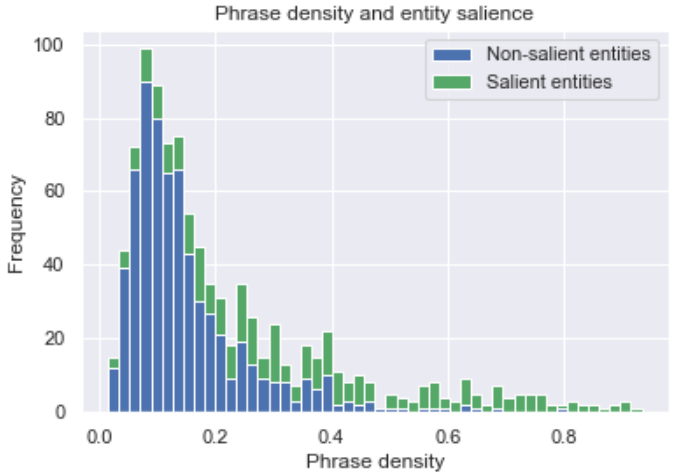

Figure 9.7: Distribution of entity salience given the phrase density.

9.2.4 Average phrase length and entity salience

Is there a relation between the average phrase length (the average number of characters in all entity phrases) and entity salience? The answer to this question is shown in figure 9.6. There is no clear relation between the average phrase length and entity salience.

9.2.5 Phrase density and entity salience

Here, phrase density is defined as the ratio between the number of words in all entity phrases and the number of words in the document. Note that the phrase density is a number between 0 and 1 where 0 means that there are no entity phrases at all and where 1 that all document words belong to an entity phrase for the given entity. The results are shown in figure 9.7. Notice that the higher the phrase density, the higher the probability that an entity is salient.

9.3 Overlapping entities

For the Wikiphrase and Wikinews datasets, the entities were already given. These entities are entities that were automatically extracted and linked to a knowledge base. Entities not linked to a knowledge base are not used in the Wikiphrase and Wikinews datasets. On the other hand, the entities extracted by the NLP toolkit (by the Named Entity Recognition (NER) component) are not linked by any linkage process. Therefore it is expected that the number of entities found by the NLP toolkit is larger than the number of entities in the Wikinews and Wikiphrase dataset. For further analysis, only entities both found in the manual annotations and extracted by the NLP toolkit are used. In total, 965 unique document-entity pairs are found in the Wikiphrase dataset, 2561 unique document-entity pairs are found by the NLP toolkit and 2925 unique document-entity pairs are found by either the Wikiphrase dataset or by the NLP toolkit. That means that the Wikiphrase dataset has a recall of 0.33 for the entities and the NLP toolkit has a recall of 0.88 for the entities. The Wikiphrase dataset has a lower recall of entities than the NLP toolkit.

9.4 Comparing Wikiphrase and PhraseNLP

This section answers the question of how phrases of Wikiphrase can be compared to entities and mentions found by PhraseNLP (a combination of a Named Entity Recognition (NER) system and a Coreference Resolution system).

9.4.1 Recall matrix

To estimate the recall, a recall matrix is computed in which each column represents a phrase found in the Wikiphrase dataset and each row represents an entity mention found by PhraseNLP. Each cell in the matrix is the number of times that the mention is found within the given phrase or the number of times that the exact entity text is found in the phrase. Consider the following example:

For the example, the following text is used:

Alice and Bob were into cryptography. Alice sent an encrypted message to Bob, but Bob lost all of his keys. Therefore, Bob was unable to read her messages. After a while, Bob managed to restore all of his keys. He was able to decrypt all the messages.

Assume that the following entity phrases are found in the text:

1. Alice and Bob were into cryptography (column 1)
2. Alice sent an encrypted message to Bob (column 2)
3. Bob was unable to read her messages (column 3)
4. Bob managed to restore all of his keys (column 4)
5. He was able to decrypt all the messages (column 5)

Furthermore, assume that the PhraseNLP system extracted the following entities and mentions:

1. Alice (row 1)
2. Bob (row 2)
3. her (row 3)
4. his (row 4)

5. He (row 5)

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \end{array} \begin{array}{c} \begin{array}{ccccc} p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \tag{9.1}$$

In this matrix, the rows represent the mentions and entities. The columns represent the phrases. It can be seen that "Alice" (m_1) is found in phrase 1 (p_1) and phrase 2 (p_2). It can also be seen that phrase 4 (p_4) contains the entity "Bob" (m_2) and the mention "his" (m_4).

By computing these matrices, it is straightforward to compute the Wikiphrase entity recall and PhraseNLP entity recall which are defined and explained in the following sections.

9.4.2 Recall matrix for PhraseNLP and Wikiphrase

With the recall matrix, it is possible to compute the ratio of phrases that contain at least one entity (found by the NER system) or one entity mention (found by the coreference system). Here, phrase recall is defined as the ratio of phrases that contain at least one entity or entity mention found by the NLP system. The PhraseNLP recall is defined as the ratio of entities or entity mentions found in at least one manually annotated phrase. The phrase recall in the Wikiphrase dataset is 0.76. That is, 0.76 of all phrases contain at least one entity or entity mention. However, that means that 0.24 of the phrases is not found by PhraseNLP but, according to the manually annotated phrases, it should refer to at least one entity. One entity reference that is annotated by a phrase but not found by PhraseNLP is "Chinese flag". This is also a hard question. In fact, "Chinese flag" is referring to "China", but it is not equal or a direct mention of the entity "China". In these cases, the phrases might contain information about the saliency of the entity "China". Other hard to find entities are abbreviations (both "U.S." and "US" were not found as entity mention for "United States"). The coreference resolution system also has difficulties in resolving commonly known aliases such as "finals" for "World Baseball Cup". An external knowledge base might improve coreference resolution systems for these cases.

The PhraseNLP recall is 0.86. That is, 0.86 of the entities and entity mentions found by the NLP system are contained within at least one phrase. That means that 0.14 of the entities and entity mentions found by the NLP system are not contained in any phrase.

9.5 Entity salience classification using Wikiphrase

In this section, entity salience classification is evaluated where entity salience classification methods (as described in the chapter on entiy salience classification methods) are applied on the manually annotated phrases of Wikiphrase. For the evaluation, cross validation with 10 folds is used and an entity salience threshold of 0.6 is used. For each of the document-entity pairs, a document vector is computed, an entity vector is computed and the difference of the document vector and the entity vector is computed. Then, a two-layered neural network with ReLU activations and 64 hidden units is used to compute the entity salience score. The scores for evaluating 10 folds on the Wikiphrase dataset, the following scores were found:

Salience class	Precision	Recall	F_1 score
Non-salient entities	0.87 (±0.07)	0.90 (±0.03)	0.88 (±0.04)
Salient entities	0.77 (±0.06)	0.71 (±0.13)	0.74 (±0.09)

Table 9.4: The precision, recall and F_1 score using the manually annotated phrases, for which the means and standard deviations are reported.

The salient entity recall as reported in table 9.4 is higher than the recall of the positional baseline. The F_1 score of the salient entities of this method is also slightly higher than the F_1 score of the positional baseline. Also, notice the improvement in the F_1 score for the non-salient entities.

9.6 PhraseNLP and entity salience classification

In this section, the results for entity salience classification are given where the phrases are estimated by using NLP methods (NER and Coreference Resolution). StanfordNLP is used for the annotations and a sentence is classified as a phrase when it contains at least one reference to the entity. The entity salience classifier is trained and evaluated on the Wikiphrase dataset. The expectation is that the automatically extracted phrases are a subset of the manually annotated phrases and therefore the system will have a lower recall of salient entities than the manually annotated phrases.

For this evaluation, the same number of folds are used for the evaluation of the manually annotated phrases. In the following sections, the results are given for the different salience classification methods.

9.6.1 PhraseNLP+ESNN classification results

In this section, the neural network classifier is trained on top of the phrases. The neural network classifier consists of two layers with 64 hidden units and it uses ReLU as activation method.

Salience class	Precision	Recall	F_1 score
Non-salient entities	0.80 (±0.04)	0.88 (±0.04)	0.84 (±0.03)
Salient entities	0.68 (±0.10)	0.53 (±0.04)	0.60 (±0.04)

Table 9.5: The precision, recall and F_1 score using the Wikiphrase dataset, for which the means and standard deviations are reported.

From table 9.5 it can be seen that the precision of the non-salient entities is higher than the precision of the non-salient entities of the positional baseline. The recall is dropped compared to the positional baseline and the F_1 increased for the non-salient entities but dropped for the salient entities compared to the positional baseline.

A better phrase-extraction mechanism would improve the scores for salient entities and is better than the positional baseline. This research also shows what information is missing in the automatically extracted phrases. One interesting research direction would be to learn how to extract phrases. The Wikiphrase dataset can be used for this purpose.

For a comparison with other methods, this method is also evaluated on the complete Wikinews dataset.

Salience class	Precision	Recall	F_1 score
Non-salient entities	0.82 (±0.03)	0.89 (±0.03)	0.85 (±0.02)
Salient entities	0.67 (±0.10)	0.51 (±0.10)	0.58 (±0.08)

Table 9.6: The precision, recall and F_1 score using the automatically extracted phrases on the Wikinews dataset, for which the means and standard deviations are reported.

9.6.2 PhraseNLP+ESRF classification results

In this section, a Random Forest classifier is used on top of the phrases. For the Wikinews dataset, the following results were obtained:

Salience class	Precision	Recall	F_1 score
Non-salient entities	0.80 (±0.05)	0.90 (±0.06)	0.85 (±0.03)
Salient entities	0.72 (±0.12)	0.53 (±0.08)	0.60 (±0.05)

Table 9.7: The precision, recall and F_1 score using the automatically extracted phrases on the Wikiphrase dataset using a Random Forest classifier, for which the means and standard deviations are reported.

As reported in 9.7, the precision, recall and F_1 score on the Wikiphrase dataset using a Random Forest classifier are slightly better than the results obtained using a Neural Network classifier.

The following table shows the results for the Random Forest classifier trained on the Wikinews dataset:

Salience class	Precision	Recall	F_1 score
Non-salient entities	0.83 (±0.02)	0.91 (±0.02)	0.87 (±0.02)
Salient entities	0.72 (±0.04)	0.55 (±0.04)	0.62 (±0.04)

Table 9.8: The precision, recall and F_1 score using the automatically extracted phrases on the Wikinews dataset, for which the means and standard deviations are reported.

The results for the Random Forest classifier as reported in table 9.8 are slightly better than the results obtained by the Neural Network classifier.

9.7 Phrase extraction

9.7.1 Results for PhraseSeq2Seq

Here, a sequence-to-sequence model is used for the phrase extraction. An entity (together with the first two sentences of the Wikipedia page of the entity) is used as input for the encoder. Then, the decoder is processing the text word by word. For each word, an attention mechanism is used to focus on relevant words in the entity information. In order to reduce the amount of overfitting, random input words of the document are removed. This is set such that 0.80 of the input data remains. Besides that, a dropout rate of 0.10 is used for the decoder. A fragment size of 60 is used. However, the numerous attempts did not lead to a working phrase extraction method. This is probably due to the fact that the number of phrases is low and the model has many weights that should be learned. In some attempts, the model was overfitting. That is; it could perfectly reproduce the train data but it was unable to work with unseen data. Augmenting the Wikiphrase dataset

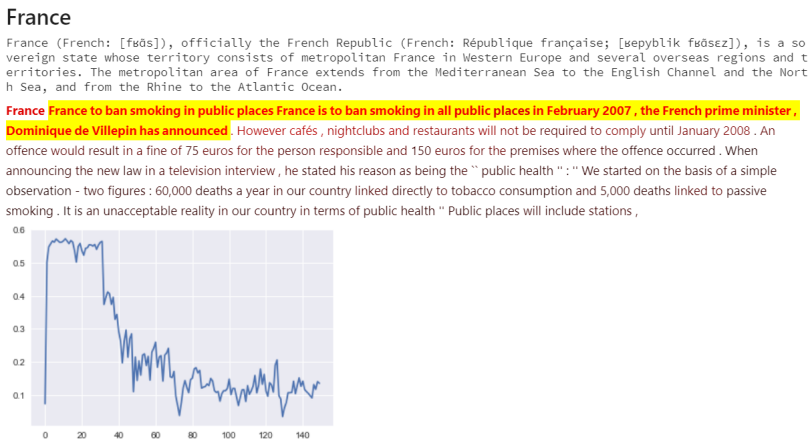

Figure 9.8: Partial results of the sequence-to-sequence phrase extraction method. The first line shows the name of the entity. Then, the first two sentences of the corresponding Wikipedia article are shown. The text highlighted in yellow is the selected phrase in the fragment (from the Wikiphrase dataset). The bold words are words selected by the phrase extraction algorithm. The redder a word, the higher the score assigned by the phrase extraction algorithm. The plot below the fragment shows the scores per word. Words with a score above 0.5 are marked as phrase words.

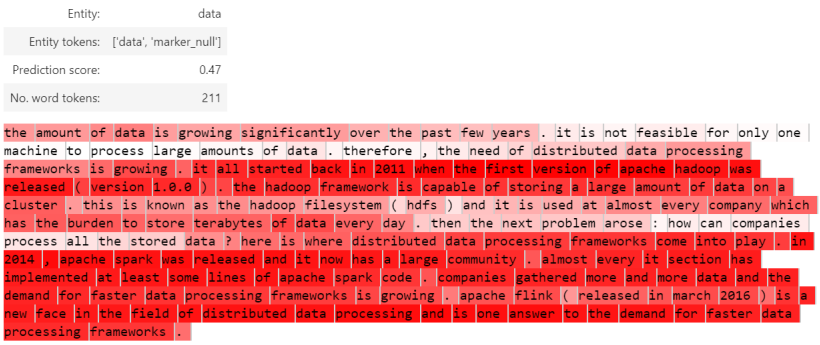

Figure 9.9: Extraction of the phrases for the entity "data" in the given document by the mESD model.

could be an option to overcome the overfitting problem. In figure 9.7.1, the partial results of the phrase extraction algorithm are shown.

9.7.2 Results for mESD

The salience classification did not work well in the mESD model. The obtained precision, recall and F_1 score were similar to the scores of a random binary classifier (computed using the formulas in the section about precision, recall and F_1 scores for a binary classification task).

On the other hand, the mESD model is capable of predicting phrases in a given document-entity pair. It is not possible to evaluate the results, since not enough phrase data is available for splitting the dataset into a train and a test set. Figure 9.7.2 shows the results obtained for a generic term ("data") and figure 9.7.2 shows the results obtained for a specific term ("Apache Flink"). However, the model is not good at predicting phrases for words which are not found in the document.

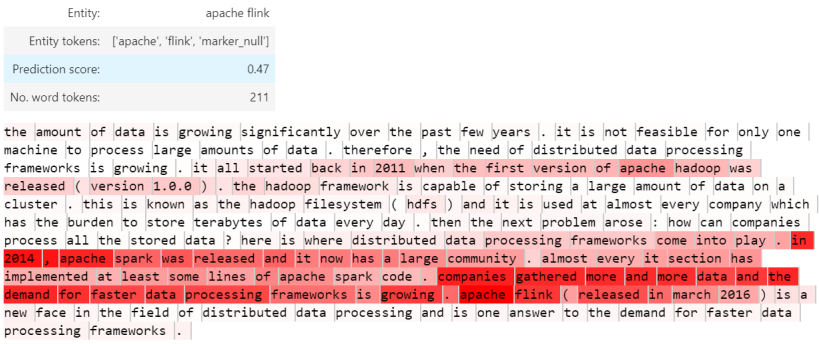

Figure 9.10: Extraction of the phrases for the entity "Apache Flink" in the given document by the mESD model.

Chapter 10

Discussion

10.1 Datasets

An issue with entity salience research is the number of publicly available datasets. This number is not large due to copyright constraints. It would help to develop larger, more variational datasets for entity salience detection which are publicly available. In this work, a small-scale, publicly available phrase and entity salience dataset (called Wikiphrase) is introduced.

10.2 Methods

One interesting research direction is the scalability of entity salience methods. It is interesting to see how this method would perform on short texts (like Tweets) and how the method would perform on long texts (like books).

The automated phrase extraction and entity salience classification system based on an NLP toolkit does work in practice and has no heavy computational steps (except for the NLP preprocessing). It might be interesting to implement a different phrase extraction system in order to speed up entity salience classification. Two attempts of phrase extraction systems are given in this work: PhraseSeq2Seq and mESD.

PhraseSeq2Seq, the phrase extraction system based on a sequence-to-sequence model did not work. One possible cause could be that the network is overfitting since there are not enough annotated phrases. One possible research direction is to increase the number of annotated phrases in the Wikiphrase dataset such that there are enough examples for an automated phrase extraction system. One other possible cause is the learnability of multiword expressions. To tackle this problem, the mESD model is created.

The mESD model is able to extract phrases. It is an end-to-end model for predicting entity salience and phrase extraction. The phrase model uses attention to relate words in the entity to words in the document and therefore, it is able to spot multi-word expressions. However, the entity salience classifier of mESD is not better than a random binary classifier. One possible future research direction is to enhance the model with a different entity salience classifiers.

The mESD model is created in the final phase of the research, and therefore it is not evaluated yet. It would be interesting to see whether augmentation of positional information would improve the entity salience classification results. It is hard to compare both the PhraseSeq2Seq model and the mESD model given the small number of labeled phrases. A larger crowdsourced entity phrase-salience dataset remains future work. This dataset can then be used for comparing entity phrase-salience models. Another research direction is to apply one of the salience classification methods on top of the phrases

extracted by the mESD model or to incorporate well-known entity salience features into the end-to-end model.

10.3 Phrases and entity salience

There are relations between entity phrases and entity salience. Just like the positional baseline, the smaller the first entity phrase position, the higher the probability of an entity being salient. And like the relation between term frequency and entity salience it holds that the larger the amount of entity phrases, the higher the probability of an entity being salient. Likewise, the larger the entity phrase density, the higher the probability of an entity being salient. And the smaller the distance between the average word vector of all phrases and the average word vector of all words in the document, the larger the probability of an entity being salient. Therefore, there are clearly features of entity phrases that can be linked to entity salience.

Chapter 11

Conclusions

Phrase-based entity salience classification improves the recall for salient entities with respect to the positional baseline when evaluated on the manual annotations of the Wikiphrase dataset. A phrase is a sequence of words expressing a relation between one or more entities or entity mentions. A phrase gives an answer to the question where relevant information is found about an entity in a given document. Therefore, phrases contain more information than entity mentions or entities themselves but phrases contain less information than sentences. In this research, the publicly available Wikiphrase dataset is constructed in which a subset of the Wikinews dataset is augmented with document-entity phrases; per entity in a given document, all phrases are given. Per document, a document vector is computed from all phrases and a document-entity vector is computed from all entity-specific phrases within the document. Using these vectors, the entity salience is estimated by a classifier and it got a higher recall for salient entities than the positional baseline (both evaluated on the Wikiphrase dataset).

Positional encodings are scalable and fast to compute encodings for the position and are useful for entity salience detection. Unlike the bucketed positions as used in [3], the positional embeddings can encode any position (as long as enough dimensions are used).

In order to automate the phrase extraction, an NLP toolkit (here: Stanford CoreNLP 3.9.1) is used. The toolkit extracts entities (by a Named Entity Recognition system) and entity mentions (using the Coreference Resolution system) and then it finds sentences containing the entity or entity mentions for a given entity. These sentences are then used as phrases and entity salience is estimated by the same classifier as used for the manually annotated phrases in Wikinews. Naturally, the precision and the recall of salient entities drops using the automated phrase extraction instead of using manually annotated phrases. Problems were abbreviations of entities which also might cause a drop in the recall. A reason for the drop in precision might be that complete sentences contain too much information.

PhraseNLP, the automated phrase extraction and entity salience classification system by using a NLP toolkit does work in practice and has no heavy computational steps (except for the NLP preprocessing). It might be interesting to implement a more precise phrase extraction system in order to speed up entity salience classification, but this remains future work.

One end-to-end model (mESD) is trained for both the entity salience detection task and the phrase extraction task. However, for the entity salience detection task it performed as good as a random binary classifier. There are straightforward future research directions for the mESD model as discussed in the discussion chapter. It could be caused by the fact that the model is not able to learn positional and frequency information of the phrases. Adding these features or enhancing the model such that it could learn these feature is an additional research direction.

The research question stated in the introduction was the following: "What is a good way to compute entity salience based on entity-attribute relations in a document and entity-attribute relations found in a knowledge graph?" One possible way is the implementation of the described phrase extraction systems. However, more phrase annotations are needed and a crowdsourced entity phrase-salience dataset is set out as future work. This larger dataset can be used to compare and create better entity phrase-salience methods. Entity information (such as entity attributes) can be encoded in an encoder module. This encoding can be updated by processing entity knowledge base fragments. The key idea of the phrases is that the phrases contain more information than single entities or entity mentions. The recall for entity salience detection is increased in a phrase-based system using the manually annotated phrases compared to the positional baseline or the described NLP-based approach. There is still room for improvement to improve on the phrase extraction task and this might lead to an improvement on the entity salience classification task.

Bibliography

[1] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.

[2] Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. Analysis of named entity recognition and linking for tweets. 51(2):32–49.

[3] Jesse Dunietz and Daniel Gillick. A New Entity Salience Task with Millions of Training Examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, Volume 2: Short Papers*, pages 205–209. Association for Computational Linguistics.

[4] Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. Centering: A Framework for Modelling the Local Coherence of Discourse,:.

[5] Vishal Gupta and Gurpreet Singh Lehal. A Survey of Text Summarization Extractive Techniques. 2(3).

[6] Kazi Saidul Hasan and Vincent Ng. Automatic Keyphrase Extraction: A Survey of the State of the Art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273. Association for Computational Linguistics.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. 9(8):1735–1780.

[8] Tibor Kiss and Jan Strunk. Unsupervised Multilingual Sentence Boundary Detection. 32(4):485–525.

[9] Patrice Lopez and Laurent Romary. HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. page 5.

[10] H. P. Luhn. The Automatic Creation of Literature Abstracts. 2(2):159–165.

[11] Daniel Marcu. Discourse trees are good indicators of importance in text. page 14.

[12] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. 5(4):115–133.

[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. page 9.

[14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

[15] Marco Ponza, Paolo Ferragina, and Francesco Piccinno. SWAT: A System for Detecting Salient Wikipedia Entities in Texts.

[16] Priya Radhakrishnan, Ganesh Jawahar, Manish Gupta, and Vasudeva Varma. SNEIT: Salient Named Entity Identification in Tweets. 21(4).

[17] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. 65(6):386–408.

[18] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. Brat: A Web-based Tool for NLP-Assisted Text Annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107. Association for Computational Linguistics.

[19] Salvatore Trani, Claudio Lucchese, Raffaele Perego, David E. Losada, Diego Ceccarelli, and Salvatore Orlando. SEL: A unified algorithm for salient entity linking: A Unified Algorithm for Salient Entity Linking. 34(1):2–29.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

[21] Shuohang Wang and Jing Jiang. Learning Natural Language Inference with LSTM. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1442–1451. Association for Computational Linguistics.

[22] Shuohang Wang and Jing Jiang. Machine Comprehension Using Match-LSTM and Answer Pointer.

[23] Paul John Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Harvard University.

[24] P.J. Werbos. Backpropagation through time: What it does and how to do it. 78(10):1550–1560, Oct./1990.

[25] Chenyan Xiong, Zhengzhong Liu, Jamie Callan, and Tie-Yan Liu. Towards Better Text Understanding and Retrieval through Kernel Entity Salience Modeling. pages 575–584.

List of Figures

List of Tables