UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Matching Red Links with Wikidata Items

Author:
Kateryna LIUBONKO

Supervisor:
Diego SÁEZ-TRUMPER

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2020

Declaration of Authorship

I, Kateryna LIUBONKO, declare that this thesis titled, "Matching Red Links with Wikidata Items" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UKRAINIAN CATHOLIC UNIVERSITY

Abstract

Faculty of Applied Sciences

Master of Science

Matching Red Links with Wikidata Items

by Kateryna LIUBONKO

This work tackles the problem of matching Wikipedia red links with existing articles. Links in Wikipedia pages are considered red when lead to nonexistent articles. In other Wikipedia editions could exist articles that correspond to such red links. In our work, we propose a way to match red links in one Wikipedia edition to existent pages in another edition. We solve this task in a context of Ukrainian red links and English existing pages.

We created a dataset of 3 171 most frequent Ukrainian red links and a dataset of 2 957 927 pairs of red links and the most probable candidates for the correspondent pages in English Wikipedia. This dataset is publicly released[1].

We defined the task as a Named Entity Linking problem. Red links are named entities and we link Ukrainian red links to English Wikipedia pages.

In this work we provide a thorough analysis on the data and define its conceptual characteristics to exploit in entity resolution. These characteristics are graph properties (connections with the pages where red links occur and connections with the pages which occur in the same pages with red links) and word properties (title names).

BabelNet knowledge base was applied to this task. We evaluated its powers in terms of F_1 score (29 %) and regarded it as a baseline for our approach. To improve the results we introduced several similarity metrics based on mentioned red links characteristics. Combined in a linear model they resulted in $F1$ score 85 % which is our best result.

In our thesis we also discuss bottlenecks and limitations of the current approach and outline the ideas for future improvements.

To the best of our knowledge, we are the first to state the problem and propose a solution for red links in Ukrainian Wikipedia edition.

All the code for this project is publicly released on github[2].

[1] https://doi.org/10.6084/m9.figshare.11550774

[2] https://github.com/Katerali/redlinks_linking

Acknowledgements

I am grateful to my thesis supervisor Diego Sáez-Trumper who supported this project with enthusiasm and contributed it with his work and advice.

Contents

List of Figures

List of Tables

List of Abbreviations

NLP	**N**atural **L**anguage **P**rocessing
EL	**E**ntity **L**inking
SDNE	**S**tructural **D**eep **N**etwork **E**mbedding
LLE	**L**ocally **L**inear **E**mbedding
synset	**syn**onim **set**

To my kind husband

Chapter 1

Introduction

Nowadays Wikipedia is constantly attracting attention of Data Scientists and Machine learning engineers. First, this multilingual encyclopedia is a subject of study per se. Secondly, it is used as a knowledge base to develop other tools (e.g. DBpedia[1], BabelNet[2]) and solve NLP problems. The proposed work addresses both ideas. It tackles the problem of gaps in Wikipedia network to remove them and at the same time exploits Wikipedia as means to do it.

Wikipedia gaps which this work refers to the ones that are caused by so called red links. The phenomenon of red links is one of Wikipedia mechanisms which have not been studied deeply yet. Red links are links to pages which do not exist (either not yet created or have been deleted). The problem of red links is that they can refer to Wikidata items or Wikipedia articles which already exists in other languages, but can not been identified from the source language. For example an article in language L_i can contain a red link to an article about A_i which does not exist in L_i, but exists in another language L_j. Our goal is to identify such connections between missing content in one language, with existing content in another language. We tackle this problem in a context of Ukrainian and English Wikipedia editions. Our solution is developed on red links of Ukrainian Wikipedia edition looking for the correspondence on the English Wikipedia edition.

Number of red links in Ukrainian Wikipedia is 1 554 986[3] unique titles. While the size of Ukrainian Wikipedia itself is 817 892 [4] existent articles. English Wikipedia is 8 times bigger than the Ukrainian edition (5 719 743 articles[5]). Therefore, the idea is to use the English version as a knowledge base to fill the gaps of Ukrainian red links.

1.1 Problem Statement

One of the fundamental characteristics of Wikipedia is that it is being constantly created. Red links is one of the mechanisms that helps Wikipedia to grow. However, the process of creating new articles by means of red links have to be really optimized and fostered. For now it is unmanageable to estimate the exact numbers of red links which have existent correspondent pages in other Wikipedia editions. However, a manual inspection clearly shows that this a frequent phenomenon. If managed appropriately red links may be better encapsulated in the Wikipedia network and faster transformed to full articles.

[1] https://wiki.dbpedia.org/
[2] https://babelnet.org/
[3] Wikipedia dumps from the 20th of September, 2018
[4] same
[5] same

Several projects in English Wikipedia community were held to tackle this problem. The most relevant for our work are Red Link Recovery Wiki Project[6] and Filling red links with Wikidata project[7]. Still they are either only discussed as an idea and not implemented or not currently maintained.

The alternative to red links in Wikipedia can be considered templates with interlanguage links (Figure 1.1). These templates are used to link important concepts in Wikipedia articles to already existent Wikipedia pages in another edition (in case of 'часткова функція' that we see on Figure 1.1 it links to English edition). Thus getting desirable information or creating new articles become more efficient. The markup for templates with interlanguage links is {{ нп|треба=часткова функція|є=Partial function}}. Red links are marked in a different way which is [[багатозначна функція|багатозначною функцією]]. In this way all links to existent articles in Wikipedia are marked which causes difficulties to distinguish red links from other links when processing the Wikipedia data.

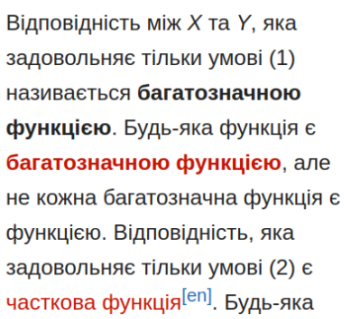

FIGURE 1.1: Examples of a red link ('багатозначною функцією') and a template[8] with interlanguage link ('часткова функція')

1.2 Our approach

We approach the problem of red links as a Named Entity Linking task (Zheng et al., 2010). The reason is that the majority of Wikipedia articles are about Named Entities and we solve these entities (red links) linking them to English articles. We use graph and word properties of Wikipedia articles and apply different similarity metrics to find the correspondent items in English Wikipedia for Ukrainian red links. Finally we compare the results of these metrics with the results of BabelNet knowledge base considering the last as our baseline. Among all the applied techniques we present the similarity model that produces the best results.

1.3 Main contributions

We consider the following to be the main contributions of our work:

- We present a solution for filling the gaps in Ukrainian Wikipedia network using the English Wikipedia edition as a knowledge base. To the best of our knowledge we are the first to tackle the problem of red links in Ukrainian Wikipedia.
- We create a dataset of 2 957 927 pairs[9] of red links and the candidate articles in English Wikipedia for the most frequent 3 171 red links from Ukrainian Wikipedia. We label it with ground truth and publicly release it[10].

[6] https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Red_Link_Recovery
[7] https://meta.wikimedia.org/wiki/Filling_red_links_with_Wikidata
[9] Wikipedia dumps from the 20th of September, 2018
[10] https://doi.org/10.6084/m9.figshare.11550774

- We present a data analysis of red links in Ukrainian Wikipedia which can foster further investigation in this field.
- We publicly release the code on github[11].
- We present the powers of BabelNet tool for Entity linking task resolving Ukrainian red links. As far as we know we are the first to apply BabelNet to such a task.

1.4 Thesis organization

The organization of this thesis is as follows. Chapter 2 describes related work on solving the red links problem by the Wikipedia community and recent advances in Named Entity Linking task. Chapter 3 gives theoretical foundations for our future model. Chapter 4 highlights the pipeline of our project including data retrieval, processing and analysis, creating dataset and forming baseline, building models and proving results. In Chapter 5 we discuss the obtained results, analyze the limitations and the problems of our approach and outline the future work.

[11] https://github.com/Katerali/redlinks_linking

Chapter 2

Related Work

2.1 Wikimedia projects and tools

To the best of our knowledge, there are not scientific publications working on matching red links to Wikidata items. Several projects were held by Wikipedia community but with no publicly published peer-reviewed papers. Among them is a Red Link Recovery Wiki Project for English Wikipedia (*Wikipedia:WikiProject Red Link Recovery/RLRL*). The community had been contributing to the project until 2017. The main goal of this project was to reduce the number of irrelevant red links. Red Link Wiki Project is of our interest because there was developed a tool Red Link Recovery Live to suggest alternative targets for red links. Although the targets were in the same Wikipedia edition the methods used there can be applied to our task as well. Some of the techniques to evaluate this similarity for Red Link Recovery Live are the following:

- Weighted Levenshtein edit distance.
- Names with alternate spellings.
- Matching with titles transliterated (from originally non-Latin entities) using alternative systems (e.g. Pinyin, Wade-Giles).
- Matching with titles spelled with alternative rules (e.g. anti personnel / anti-personnel / antipersonnel).

There was also a project proposal in Wikimedia community called Filling red links with Wikidata which intention was to make red links a part of a Wikipedia graph (*Filling red links with Wikidata, Wikimedia Meta-Wiki*). The aim is similar to ours but it is related with the particular moment of creating a red link. Its idea is to create placeholder articles filled with data from Wikidata. This project proposal has a wide perspective not only connecting red links to Wikidata items but also automatically creating Wikipedia pages. However it was not implemented. The discussion on that project involved many questions on how to maintain and edit these new 'semi-articles'.

Also the suggestion to connect red links to Wikidata items appeared in Wiki-research-l Digest, Vol 157, Issue 19 (*Wiki-research-l Digest, Vol 157, Issue 19*) by one of the Wikimedia users and contributors Maarten Dammers. Issues on technical implementation of this idea were discussed such as creating a new property on Wikidata to store the name of the future article, hovering of the link to get a hover card in user's favorite backup language etc. These suggestions were related to the process of connecting red links to the appropriate Wikidata items when creating them. And the project was not implemented.

Languages	284
Babel synsets	15 780 364
Babel senses	808 974 108
Babel concepts	6 113 467
Named Entities	9 666 897
Images	54 229 458
Sources	47

TABLE 2.1: General Statistics of BabelNet 4.0

Another work that is of our interest is Improving Website Hyperlink Structure Using Server Logs (Paranjape et al., 2016). In this work Wikimedia server logs are used to predict which links are needed to make Wikipedia graph more complete. They build navigation trees where a size is a number of page views. Then candidates for new links are created and a clickthrough rate for each candidate is estimated. They propose several methods of estimation and then based on these results build their predictions. This work is interesting for us as presents methods to work with Wikipedia as a graph. Moreover it introduces server logs as a feature which can even be applied further in our work to rank red links by their importance.

The projects described above are all in the domain of English Wikipedia edition. For Ukrainian edition the only thing that was found related to the red links problem is gathering lists of red links and combining them into topics.

We also explored the tools and resources developed by Wikimedia research engineers and volunteers that can be useful for solving our task. The first is a powerful tool called PetScan (*PetScan tool for Wikimedia*). It helps to obtain information on red links with a user interface. It is developed by Wikimedia Toolforge (*Wikimedia Toolforge for developers*) which is a hosting environment for developers working on services that provide value to the Wikimedia movement. There we could find more information for our work.

Another Wikimedia tools and resources for research we used are Wikimedia Static Dumps[1], SQL Replicas[2] and Quarry[3]. SQL Replicas is a 'set of live replica SQL databases of public Wikimedia Wikis'. And Quarry is a 'public querying interface for Wiki Replicas'. It is the first place to get a structured Wikipedia data. Wikimedia Static Dumps are a 'complete copy of all Wikimedia wikis, in the form of wikitext source and metadata embedded in XML'. Dumps will serve us to extract the necessary data which is absent in SQL Replicas.

2.2 BabelNet

We considered a work by a research community from Sapienza University of Rome to be relevant for our task. They implemented a knowledge base (*BabelNet 4.0 and Live Version*) which serves as a multilingual encyclopedic dictionary and a semantic network. BabelNet is initially constructed on Wikipedia concepts and WordNet[4] database. The main idea behind it is that encoding knowledge in a structured way

[1] https://dumps.wikimedia.org/

[2] https://upload.wikimedia.org/wikipedia/commons/9/94/MediaWiki_1.28.0_database_schema.svg

[3] https://quarry.wmflabs.org

[4] https://wordnet.princeton.edu/

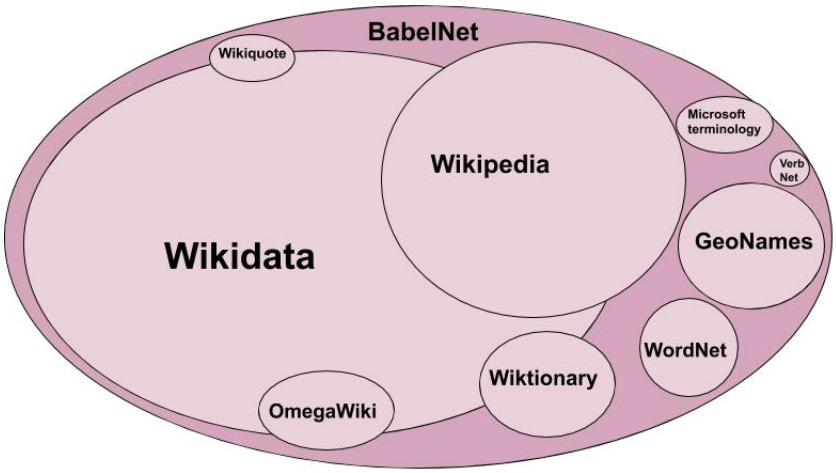

FIGURE 2.1: Principal components of BabelNet 4.0 (English version)

helps to solve different NLP tasks even better than statistical techniques. The latest BabelNet 4.0 version contains data from 47 sources (mainly from OmegaWiki[5], VerbNet[6], GeoNames[7], Semcor[8] automatic translations etc.). Its power is different for different languages as each one has a particular amount of supporting sources. The most powerful is obviously English. Nowadays BabelNet contains knowledge bases for 284 languages. These knowledge bases include not only lexicalized items but also images. General statistics on the last version of BabelNet is presented in table 2.1 and its main constituents are presented in figure 2.1.

The core concept of BabelNet is a synset which is deciphered as a synonym set. It is a set of synonyms in multiple languages for a word particular meaning. For example for a word 'play' in the meaning of a dramatic work intended for performance by actors on a stage there is a multilingual synset (play, Theaterstück, dramma, obra, . . . , pièce de théâtre). On the other hand for a word 'play' in the meaning of a contest with rules to determine a winner the synset is (UTF8gbsn , game, jeu, Spiel, . . . , juego). For this reason BabelNet can tackle the ambiguity problem.

BabelNet had not been applied to a red links problem in Wikipedia before. The closest for our project task where BabelNet's power was tested is a Multilingual All-Words Sense Disambiguation and Entity Linking (Moro and Navigli, 2015) for SemEval-2015 Task 13. Thanks for its content and structure BabelNet showed high results for finding the correct translations for multi-meaningful words, especially it worked well for nouns and noun phrases which make the majority of Wikipedia titles.

[5] http://www.omegawiki.org/Meta:Main_Page
[6] https://verbs.colorado.edu/verbnet/
[7] https://www.geonames.org/
[8] https://www.semcor.net/

2.3 Entity linking

Named Entity Linking is a task to map a named entity mentioned in a text to a corresponding entry stored in the existing knowledge base (Zheng et al., 2010). Named Entity Linking task is also called Named Entity Disambiguation process as by linking named entity to a particular concept in a knowledge base we disambiguate it from other concepts. Knowledge bases used for Entity Linking tasks are mainly Wikipedia and its subsequent projects such as DBpedia, Wikidata[9], YAGO[10] (Fabian, Gjergji, and Gerhard, 2007), Freebase (Bollacker et al., 2008). The reason to use them is that currently they are the fullest and all-encompassing networks of knowledge.

Entity linking serves for information retrieval tasks such as creating text summary, search engines, also helps with augmenting text with links and so on.

The theoretical basis for Entity linking pipeline we derived from the work (Shen, Wang, and Han, 2014) as they present a good overview of main approaches to entity linking. There the general modules in entity linking pipeline are described as following:

1. Candidate Entity Generation

   In this module, for each entity mention m $\in$ M, the entity linking system aims to filter out irrelevant entities in the knowledge base and retrieve a candidate entity set E_m which contains possible entities that entity mention m may refer to (Shen, Wang, and Han, 2014).

2. Candidate Entity Ranking

   In most cases, the size of the candidate entity set E_m is larger than one. Researchers leverage different kinds of evidence to rank the candidate entities in E_m and try to find the entity e $\in E_m$ which is the most likely link for mention m (Shen, Wang, and Han, 2014).

3. Unlinkable Mention Prediction

   To deal with the problem of predicting unlinkable mentions, some work leverages this module to validate whether the top-ranked entity identified in the Candidate Entity Ranking module is the target entity for mention m(Shen, Wang, and Han, 2014).

Next, they describe various approaches to construct a name dictionary which is essential for the Entity Linking pipeline. Name dictionary is a set of key value pairs where keys are entity names and values are different concepts which can refer to these names. For example, entity concept Michael Jordan (key) would have names (values) such as: Michael Jordan, Michael I. Jordan, Michael Jordan (footballer), Michael Jordan (mycologist). Entity disambiguation is reached by ranking these values and mapping values to proper keys.

The first benefit for us from this work is that it presents the basic schema how to approach our problem of linking red links to English Wikipedia. The second is that it exploits Wikipedia properties for Entity Linking which is also suitable for our task.

[9] https://www.wikidata.org/wiki/Wikidata:Main_Page

[10] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

2.4 Embedding for concept similarity

One of the mainstream techniques in current NLP approaches used for Entity Linking tasks in particular is embedding.

The concept of embedding has roots in topology, differential geometry and category theory (Lee, 2013). In these fields 'embedding' means map from domain X into co-domain Y $f : X \to Y$. This map is injective (each $y \in Y$ has only one corresponding $x \in X$) and structure preserving. The kind of structure preserved in mapping depends on X and Y.

In context of Natural Language Processing, domain of the mapping X is usually words, sentences or texts and co-domain Y is a vector space. In this case 'structure preserving' means notion of a distance: in domain X (words) it is a semantic distance and in Y (vector space) it is a geometric distance between vectors.

In other words, embeddings are mathematical procedure to take a set in one domain (e.g. words) and project it in co-domain (e.g. vector space) preserving a notion of distance (e.g. semantic distance is preserved as a geometric distance).

2.4.1 Word embeddings

In word embedding a semantic similarity is preserved by relative distance between word representations in a vector space. In our work we are interested in achievements in this field with regard to our task of Named Entity Linking and to our research domain which is Wikipedia.

Applying embeddings for Named Entity Linking task, (Zwicklbauer, Seifert, and Granitzer, 2016) propose an effective graph-based algorithm which exploits embeddings on two levels, a word level and a document level. Thus they capture different semantic characteristics of entities, the meaning of a word and a meaning of the context it is used in (e.g. topic). These two features help them to achieve better than state-of-the-art results in entity disambiguation task. Moreover, they call their algorithm a robust one as it works for different data.

In work (Sherkat and Milios, 2017) each Wikipedia article is embedded as a separate concept. Authors present their results in Concept Analogy and Concept Similarity tasks. These experiments on embedding Wikipedia pages gave us an understanding of embedding possibilities in terms of Wikipedia and some ideas for further research.

At another point, FastText[11] is releasing word embeddings trained on Wikipedia. It is an open-source library which is widely used for research nowadays.

2.4.2 Graph embeddings

The main idea of graph embedding is to create a new vector representation of a graph preserving the similarity among nodes. As mention (Parravicini et al., 2019), 'while text-based embeddings have shown good results, use of vertex embeddings offers more flexibility with respect to the field of application, as no large text corpus is required to create the embeddings'. With their algorithm, they 'implement and evaluate a reference pipeline that uses DBpedia as knowledge base and leverages specific algorithms for fast candidate search and high-performance state-space search optimization'. Except high accuracy results the advantage of their work is that it is performed in a real-time.

[11] https://fasttext.cc/

In our thesis project we appeal to a work of (Goyal and Ferrara, 2018), both theoretical part and the developed python library for graph embedding. The library embraces different kind of algorithms for graph embedding. They all are grouped into factorization based, random walk and deep learning based methods. This companion open-source Python library is called GEM (Graph Embedding Methods) and can be found on githubḟootnotehttps://github.com/palash1992/GEM

Thus we see the following state of the related work on matching red inks to Wikidata items. There are no papers on matching red links to Wikidata items known for us. Red links in Ukrainian Wikipedia edition have attracted little attention. Work on reducing red links had been carried by a WikiProject 'Red Link Recovery' from 2005 to 2017 but it concentrated on finding existent articles for red links in the same edition. Project proposals concerning red links problems were made within Wikimedia community. Powerful tools are provided by Wikimedia Foundation which are useful for information retrieval on Wikipedia items. A potent knowledge base BabelNet was developed which may solve matching red links to existent pages in Wikipedia but the tool was not yet applied to this particular problem. This work can be used in our Master project either partially or as ideas for further work and for further applications of our model within Ukrainian Wikipedia.

In its turn, in the field of Entity linking task there is a developed methodology both in general and in the context of Wikipedia. Also some techniques such as word and graph embeddings are highly developed and can be brought on trial in our task.

Chapter 3

Similarity-based Model for Entity Linking Task

In this section we present our solution for red link matching problem. We introduce two macro-approaches which help to catch different features of the data. The first is based on graph properties and the second is based on word properties. Then we combine them using a linear model.

3.1 Named Entity Linking Task

Based on previous work we propose to use the following components of a Named Entity Linking pipeline:

1. **Similarity metrics.** That is a comparison method which uses particular characteristics of items and particular rules to estimate comparison score.
2. **Ranking rules.** They imply defining a certain threshold or choosing top X results to mark as true.
3. **Ground truth.** For each named entity a ground truth should be known in order to make final evaluation of the model.
4. **Evaluation metrics.** It should be a universal metrics appropriate to the task.

All these components are chosen with regard to data in form of prepared data sets: a set of named entities to link and a set of candidates for each named entity to link with.

3.2 Similarity based on Graph properties

Representing Wikipedia as a graph we refer to its elements as nodes and links between nodes. We see the Wikipedia graph in the following way:

- Nodes of the Wikipedia graph are existent articles and red links;
- Incoming link with regard to an article is a link to this article. In Wikipedia it means that the article is mentioned in another article;
- Outgoing link with regard to an article is a link from this article. In Wikipedia it means that the article mentions another article;
- concurrent links are outgoing links from the same article. In Wikipedia it means that articles occur in the same Wikipedia article.

We can represent it as in figure 3.1 Here F_1, F_2 and F_3 are existent Wikipedia articles. R is a red link. The specifics of this graph is that all links are directed. Nodes which represent existent articles can have both incoming and outcoming links as nodes F_1 and F_2.

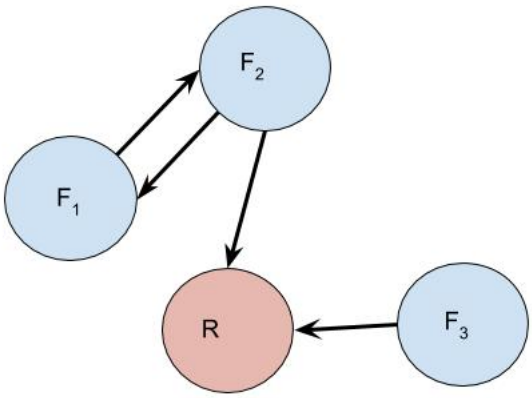

FIGURE 3.1: Representation of a part of a Wikipedia graph

Red links articles do not have outcoming links. They can be described by incoming and concurrent links. In figure 3.2 the incoming links for a red link R are from nodes F_2 and F_3.

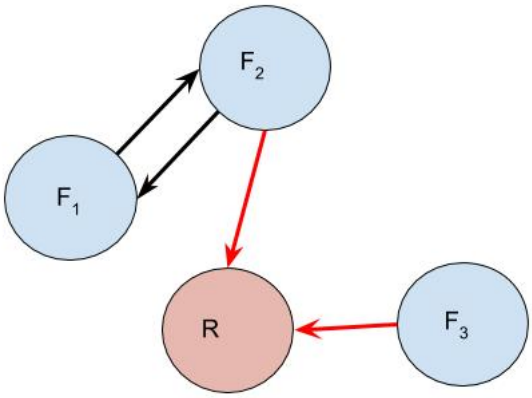

FIGURE 3.2: Incoming links for a red link

Concurrent link for the red link R in this case is the edge from node F_2 to node F_1 (figure 3.3).

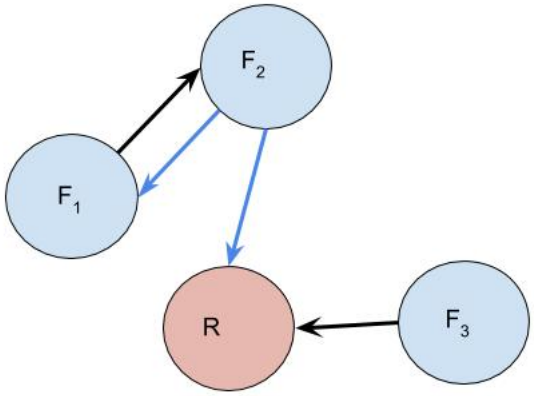

FIGURE 3.3: Concurrent links for a red link

3.2.1 Calculating links in common with Jaccard similarity

As follows from the characteristics of red links, incoming and concurrent links can be used as their features by which we compare them with other Wikipedia articles. Similarity measure we use is Jaccard score (Kosub, 2019). It is applied to compare sets which are unordered collections of objects. The idea behind it is to calculate the fraction of common elements in the considered sets over all the elements of these sets. In terms of sets it is defined as an intersection over union and formalized in the following way:

$$S_{AB} = \frac{A \cap B}{A \cup B}, \tag{3.1}$$

where A and B are two sets to compare.

Jaccard similarity metrics results in a number within the margin from 0 to 1, where 0 means no similarity and 1 means totally similar.

Common incoming links

In case of comparing articles by incoming links we are going to represent each article (that is each node) as a set of its incoming links. Then our formula will be the following:

$$S_{I_{eng}I_{ukr}} = \frac{I_{eng} \cap I_{ukr}}{I_{eng} \cup I_{ukr}}, \tag{3.2}$$

where I_{eng} is a set of incoming links for an English non-translated article and I_{ukr} is a set of incoming links for a Ukrainian red link. Thus we can compare articles by pages where each one occurs.

Common concurrent links

In case of comparing articles by concurrent links we are going to represent each article as a set of its concurrent links. Then our formula will be the following:

$$S_{C_{eng}C_{ukr}} = \frac{C_{eng} \cap C_{ukr}}{C_{eng} \cup C_{ukr}}, \tag{3.3}$$

where C_{eng} is a set of concurrent links for an English non-translated article and C_{ukr} is a set of concurrent links for a Ukrainian red link.

Thus we can compare articles by articles which occur in the same page.

3.2.2 Graph embedding

For this problem the theoretical and software background was based on the paper of Palash Goyal and Emilio Ferrara (Goyal and Ferrara, 2018) and their library GEM 5[1]. Among different embedding techniques described in that article and implemented in the library we have chosen Locally Linear Embedding and Structural Deep Network Embedding (SDNE). The choice of Locally Linear Embedding was due to way it embedded the nodes – it assumes that every node is a linear combination of its neighbors in the embedding space. SDNE was chosen due to its good results in experiments provided by authors of the article.

3.3 Similarity based on Word properties

3.3.1 Levenshtein edit distance

Levenshtein distance is one of the best approved metrics to measure the similarity between two sequences of symbols. 'This measure is often called the "edit distance" and can be defined as the minimum cost of transforming one string into another through a sequence of weighted edit operations.' (Yujian and Bo, 2007) The transformation operations are deletion, insertion and substitution.

A formal definition of the Levenshtein distance (introduced by Dan Jurafsky using concepts of dynamic programming (Jurafsky and Martin, 2019)) is

$$D[i,j] = min \begin{cases} D[i-1,j] + del - cost(source[i]) \\ D[i,j-1] + ins - cost(target[j]) \\ D[i-1,j-1] + sub - cost(source[i], target[j]) \end{cases} \tag{3.4}$$

Here source[i] is a position of a character in a source string (which we compare) and target[j] is a position of a character in a target string (to which we compare).

There can be slight modifications for this edit measure. First, each operation (insertion, deletion, substitution) can be weighted differently. In our case we leave it with default uniform weights, each operation costed 1. Second, minimum edit distance can be applied in Generalized Levenshtein Distance form or be normalized. The reason of normalizing is that we have sequences of different sizes and 'two errors in a comparison of short strings are more critical than in a comparison of long strings.' (Yujian and Bo, 2007) In our project we refer to edit distance normalized by the longest string among a red link and a candidate.

This metrics results in a number within the margin from 0 to 1, where 0 means the items are the same (edit distance is 0) and 1 means totally different.

3.3.2 Cross-lingual Word Embedding

Nowadays cross-lingual word embedding is a technique which answers the need of cross-lingual research and applications for modern global community. It is a transfer of monolingual word embedding techniques (which is a vector representation of words in a linear space) into the context of several languages. For that a notion of a joint embedding space is introduced. Two main reasons of using cross-lingual embeddings are highlighted by (Ruder, Vulić, and Søgaard, 2017). First, they enable us to compare the meaning of words across languages, which is key to bilingual lexicon

[1] https://github.com/palash1992/GEM

induction, machine translation, or cross-lingual information retrieval, for example. Second, cross-lingual word embeddings enable model transfer between languages, e.g., between resource-rich and low-resource languages, by providing a common representation space.

In our task we base on fastText library for learning text representations as it is trained on Wikipedia corpora. As (Ruder, Vulić, and Søgaard, 2017) conclude from their research, the data a method requires to learn to align a cross-lingual representation space is more important for the final model performance than the actual underlying architecture.

Together with Babylon multilingual project[2] fastText is employed to create a tool for mapping word meanings for 78 languages. Vector representations for Ukrainian words are trained as well.

With this tool we appeal to a mapping-based approach of cross-lingual embedding. This method consists in training word embedding separately in different languages and then align them using some dictionary. Then a transformation matrix to switch between spaces is searched. With this matrix cross-lingual tasks are performed.

3.4 Multi-factor Similarity-based Model

Based on defined properties (graph similarity and word similarity) we can apply different models to solve our task. Those properties could be considered as factors (features) and the problem could be formulated using standard machine learning concepts. We will treat the problem as supervised modeling, where one instance is a collection of obtained properties and label is whether a candidate is the actual correspondent page to a red link or not [3]. With such settings we have binary classification problem.

Taking into account the fact that there is only four features we will concentrate on linear model for binary classification. Linear model is simple and robust and can serve as a starting point for future modeling. Moreover, results are highly interpretative, given a clear notion of features importance.

Logistic regression is a model for binary classification with linear decision boundary (Bishop, 2006). The model predicts the posterior probability of one class C_1 ('true') based on a feature vector $\boldsymbol{\phi}$:

$$p(C_1|\boldsymbol{\phi}) = \sigma(\mathbf{w}^T\boldsymbol{\phi}) = \sigma(w_0 + w_1 \cdot \phi_1 + ... + w_n \cdot \phi_n), \tag{3.5}$$

where $\sigma(x) = 1/(1+exp(-x))$ is the logistic sigmoid function, n is a size of a feature vector (4 in our case), ϕ_i are features themselves, w_i are parameters of the model. The probability of a second class ('false') is obtained by $p(C_2|\boldsymbol{\phi}) = 1 - p(C_1|\boldsymbol{\phi})$.

Based on a maximum likelihood we obtain the procedure of training this model.

[2] https://github.com/babylonhealth/fastText_multilingual

[3] See Chapter 4.1.3 for details of dataset formation

Chapter 4

Experiments

In this Chapter we describe the process and results of each step in Entity Linking pipeline we undertake. In our case the named entities to link are Ukrainian red links and the knowledge base to link with is the English Wikipedia edition. Data we apply our similarity models to is Wikipedia articles of Ukrainian and English editions from September, 2018.

4.1 Data collection & pre-processing

The specifics of this work is that no prepared data and ground truth was available from the beginning. Thus we created it on our own on the basis of Wikipedia XML dumps, a langlinks SQL dump and a Wikipedia pages network. Wikipedia XML dump is a Wikipedia database backup of a certain version (time) and a certain edition (language). Langlinks SQL dump contains Wikipedia interlanguage link records. The dumps we processed contain Wikipedia data of the version by the 20th of September.

4.1.1 Data retrieval and pre-processing of the whole dataset

Our goal was to obtain red links of Ukrainian Wikipedia edition and all the corresponding information that would help to solve our matching problem. Data retrieval and some parts of pre-processing were done based on the work of our team for Mining Massive Datasets course project at Ukrainian Catholic University on Summer 2018 (*Final project for the Mining Massive Datasets course at the Ukrainian Catholic University, 2018*).

The outstanding characteristics of the input data is its size. The size of English Wikipedia is 28.0 GB in compressed format. It contains 5 719 743 full English articles. Whereas Ukrainian dump's size which we took as an input is 2.1 GB. It contains 817 892 Ukrainian Wikipedia articles of full size. The special approach was required to process this data on one computer. Mostly we split it into chunks and processed them one by one.

At first we retrieved the whole Ukrainian Wikipedia graph of pages, the whole English Wikipedia graph of pages and language links between Ukrainian and English Wikipedia graphs. From these datasets red links were obtained and other supporting datasets were formed.

Eventually we obtained 2 443 148 red links in Ukrainian Wikipedia among which 1 554 986 are unique titles.

For further matching red links to Wikidata items English Wikipedia data was processed. Thus from English Wikipedia XML dump and langlinks SQL dump we retrieved all non-translated English articles, the correspondences between Ukrainian

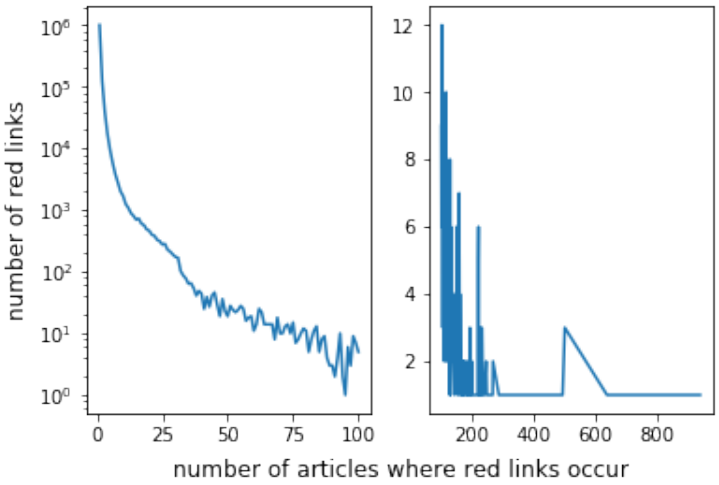

FIGURE 4.1: Frequency of occurring red links in Ukrainian articles which have langlinks to English Wiki. *Left* number of red links in 0 to 100 articles (noted the log scale). *Right* number of red links in $>$ 100 articles.

and English articles and all the incoming links to non-translated articles in English Wikipedia. The number of articles not translated to Ukrainian in English Wikipedia is 5 264 607 which means that only 8 % of English Wikipedia is translated into Ukrainian. And vice versa the number of langlinks between Ukrainian and English Wikipedia is 599 636 which is 73 % of all Ukrainian Wikipedia articles. Moreover we kept links between all Ukrainian and English Wikipedia editions to use it further in our model. For English Wikipedia it is 161 017 765 links between pages and for Ukrainian Wikipedia – 22 693 778 links.

In the Figure 4.1 we can see the frequency of how red links occur in Ukrainian articles which have langlinks to English edition. If put into numbers, there are 1 010 955 red links which occur only once. The most frequent link is 'ацетилювання'. It occurs in 941 articles. The tendency here is not linear.

4.1.2 Retrieving the sample

For the reason that we can't obtain the ground truth for such amount of red titles we decided to work for our project with samples. Thus we obtained a sample of 3 171 red titles which were in Ukrainian Wikipedia by the 20th of September 2018. The sample was obtained by choosing red titles that occur in 20 or more articles which have corresponding articles in English Wikipedia (the langlinks mentioned above). Therefore the chances to match a red link with an article from English edition are higher for this sample and through their popularity they may be more needed in Wikipedia.

Characteristics of the obtained sample: 3063 items of the considered red links are Proper Names which is 96 % of the sample. They include names of people, animal species (mostly moths), plant species, sport events, names of publishing houses, media sources, geographic locations and territories (mostly French regions), names of sport clubs, airports, administrative institutions, cinema awards and a few other minor name categories.

Among these Proper Names there are 969 red links for people's names which is 30 % of the sample. The biggest group of these names belong to tennis players.

Interestingly, a great part of red links in Ukrainian Wikipedia (at least as represented by our sample) are not in Ukrainian and many are spelled in other then Cyrillic script. The represented languages are English (e.g. 'John Wiley Sons'), Russian (e.g. 'Демографический энциклопедический словарь'), Latin (e.g. 'Idaea serpentata') and Japanese. Moreover there are 988 red links spelled in Latin script which is 31 % of the sample. Among them there are red titles in English, Latin and Ukrainian spelled in Latin script.

The data also has some innate characteristics which were obstacles for retrieval and pre-proccessing steps and which we had to take into account while building our model.

The first is double redirections – redirection pages which redirect to more redirections. For example a page'Католицизм' redirected to'Католицтво' which in turn redirected to'Католицька церква' (The only existent article here). Fortunately these double redirections are constantly checked and cleaned by Wikipedia users or bots. By the time of writing these lines the redirections mentioned above were already removed and all of them redirected directly to the full article 'Католицька церква'.

The second type of noise in data is typos in the red link titles. For example 'Панчакутек Юпанкі' is really 'Пачакутек Юпанкі', 'Сувалцьке воєводство' must be 'Сувальське воєводство'. It also goes for other mistakes in writing red links (e.g. 'Неґрська раса' instead of 'Негроїдна раса'). The dangerous thing here is that articles for these red links really exist in the Ukrainian Wikipedia but are not recognized because of the typos. Such 'false' red titles were revealed during the creation of the ground truth. Still for our current model of matching red titles to Wikidata items they have no bad impact and do not influence the model. Nevertheless this fact should be taken into account in further research.

4.1.3 Candidate pairs generation

This step is based on the work of our team for Mining Massive Datasets course project at Ukrainian Catholic University on Summer 2018 (Final project for the Mining Massive Datasets course at the Ukrainian Catholic University, 2018)

For each red link of our sample we have retrieved a set of articles from English edition which is more probable to contain an entity a red link refers to. Thus it is called a candidate set and this phase in a pipeline is called a Candidate Entity Generation. Our approach to candidate generation is based on common links comparison. The measure of similarity chosen is Jaccard score.

As an input data for retrieving future candidates we use English Wikipedia articles which do not have correspondent pages in Ukrainian Wikipedia yet. For that we process Wikimedia langlinks dump which contains data about interlanguage connections of different Wikipedia editions, in our case English and Ukrainian ones. Then from English pagelinks we get the list of all articles where these English articles are mentioned. In terms of a graph these 'parent' pages are incoming links for our future candidates. In its turn, incoming links for red articles are taken from Ukrainian pagelinks. Among those incoming links a subset of their correspondences from English Wikipedia is taken from the langlinks table. Then these incoming correspondent English links for red articles are counted.

red link	candidate
Емад Мотеаб	Mengistu Worku
Емад Мотеаб	Luciano Vassalo
Емад Мотеаб	Wael Gomaa
Емад Мотеаб	Mudashiru Lawal
Емад Мотеаб	Ali Bin Nasser
Емад Мотеаб	Emad Moteab
Емад Мотеаб	James Pritchett (footballer)
Емад Мотеаб	Federation of Uganda Football Associations

TABLE 4.1: Generated candidate pairs. Part

Based on the previous results we calculate Jaccard score similarity between red links and each of non-translated to Ukrainian English articles according to this formula

$$S_{EU} = \frac{E \cap U}{E \cup U} \tag{4.1}$$

where E is a set of incoming links for English non-translated articles and U is a set of incoming links for Ukrainian red links. With this approach we obtain the similar articles to our red links ranked from the most similar according to the Jaccard similarity score. All the links which have 0 similarity score with a red link are dropped. Thus we obtain an array of tuples (score, candidate) for each red link. Then we choose pairs which have more than 20 incoming links in common to create a sample of the most popular red links. This way a table of pairs red link – candidates is built.

A part of these pairs is given in table 4.1

A size of this set is 2 957 927 red link-candidate pairs for 3 171 red links.

This dataset is unbalanced as for each red link there are about 1 000 candidates among which either only one true candidate or no true candidate is present. It leads to particular ways of managing it in further research.

4.1.4 Creating ground truth

Ground truth for the red link data sample was not provided and no automatic tools for getting it was available. That is why the process of forming the list of correct correspondent English articles was undertaken manually in several ways. Ukrainian red link title was searched through Wikipedia and Google search engines. If no proper results were given we translated the title in Russian, English or French and repeated the search. The results could be the following

1. English Wikipedia article that we searched.
2. Corresponding Wikipedia pages in other languages where langlinks showed whether English page exists.
3. Wikidata item which contains a list of Wikipedia articles in different editions.
4. Wikispecies dictionary which also gives a link to English Wikipedia if it exists.

In the process of creating the ground truth for the sample we faced other specific features of the dataset that made the evaluation more difficult. They are the following:

- Different names for one concept or person (e.g. 'Білозубкові' and 'Білозубки'). It also leads to the articles that already exist in the considered Wikipedia edition.
- Ambiguity. It is hard to find the right correspondence to a red title just by the name (e.g. 'Austin', 'Guilford', 'Йонас Свенссон'). In this context it is often useful to point to a disambiguation page. And evidently more information than just a title is required for matching.
- Red links which by the time of checking for ground truth already became full articles in the considered edition.
- Correspondences that were found by the time of checking for ground truth became deleted articles.

Thus we conclude that the methods to deal with huge amount of data should be applied and one way is to take representative samples. And due to Wikipedia nature and structure, people's mistakes, nature of the language itself and inner nature of the relations between Ukrainian and English Wikipedia editions the considered data has some specific characteristics that should be considered when building a model.

4.1.5 Train and Test sets

For further work with our data we split it into train and test sets.

As our dataset is unbalanced we approach the splitting not in a usual way. A usual random split would divide candidates for same red links between a train and a test set. As for each red link we have at most one true candidate such kind of split would be inappropriate. Therefore we randomly split a dataset of red links (1 171 items) in fraction 80 % for a train set and 20% for a test set. Then we combine these red links with their candidates. Thus we obtain a train set of 2 337 270 pairs and a test set of 620 657 pairs.

4.2 Evaluation metrics

To evaluate and compare the results of different similarity metrics we use F-measure.

The reason we have chosen F-measure is that our dataset is imbalanced, for several thousands candidates there is only one true item. F-measure is calculated on the basis of precision and recall. Precision and recall are defined in terms of true positives (TP), false positives (FP) and false negatives (FN). True positive is an item which is true and marked by a model as true. In our case true positive is an article in English Wikipedia which corresponds to the red link we consider and is marked as such. For example, for the red link 'Анкона (футбольний клуб)' the correspondent English article 'U.S. Ancona 1905' is chosen and it is right. False positive is an item which is false but marked as true. In our case it is an article in English Wikipedia which does not correspond to the red link we consider but is marked as such. For example, for the red link 'Анкона (футбольний клуб)' the correspondent English article 'Barbara Schett' is chosen and it is false. False negative is an item which is

true but marked as false. In our case it is an article in English Wikipedia which corresponds to the red link we consider but is marked as not be a such. For example, for the red link 'Анкона (футбольний клуб)' the correspondent English article 'U.S. Ancona 1905' is not chosen but it is right.

Precision shows whether the items chosen as true are true. It is defined as

$$P = \frac{TP}{TP + FP}$$

Recall reveals whether all true items are chosen. It is defined as

$$R = \frac{TP}{TP + FN}$$

F-measure is a **harmonic** mean of precision (P) and recall (R). In its full form it is defined as:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta \cdot P + R},$$

where β is a parameter that controls a balance between P and R (Sasaki, 2007).

For our project we weight precision and recall equally which means $\beta = 1$ and the formula becomes

$$F = \frac{2PR}{P + R}$$

This kind of F-measure with equally balanced precision and recall is called F_1 score. In our work we refer to this particular evaluation metrics.

4.3 Similarity Metrics

Here we present the results of the applied similarity metrics to our data. First we present our BabelNet baseline and then compare the results of our similarity metrics with the baseline. The metrics are the following: Jaccard similarity measure on incoming and concurrent links of parts of Wikipedia graph, similarity of embedded graphs in Ukrainian and English Wikipedia, Levenshtein edit distance between red links titles and candidate article titles, cosine similarity between embedded titles of Ukrainian red links and English candidate titles. Also we experiment on how these independent similarity metrics can work together.

4.3.1 BabelNet baseline

We first applied BabelNet multilingual encyclopedia for searching correspondent articles for red links in other Wikipedia editions. BabelNet provides its API for queries and is also available online. We used both BabelNet Java API and online dictionary to ensure the correctness of results.

The process of search was the following: for a Ukrainian red link title a translation from English BabelNet dictionary was queried. In particular we queried the English Wikipedia part of the BabelNet knowledge base.

The results were evaluated through the dataset of unique red links. Thus the effectiveness of BabelNet for this task is 29 % F_1 score with precision 97% and recall 17 %. Better results BabelNet shows for the reverse task of searching correspondent Ukrainian articles for English red links. F_1 score there is 61 %. These results are obtained due to precision (89 %) and recall (46 %). It means that if any found in BabelNet it was correct but BabelNet could not find translations for 54 % of existed

items in Ukrainian Wikipedia. The main reasons of so called false negatives were the following:

1. Red link title does not exist in English BabelNet.
2. Version of BabelNet is older than a wanted page.
3. Title of a red link is different from a title of a page (different surnames but the same person).
4. BabelNet doesn't manage with typos.
5. Red link titles are often written in other than English languages. So first translation to English is needed.

In both cases (Ukrainian red links and English red links) BabelNet is good to disambiguate if contains an item corresponded to a red link. However this number is quite low: 46 % for English red links and 17 % for Ukrainian red links. There is a big field for improvement for this task especially for Ukrainian red links problem. Thus we suggest it as our baseline.

4.3.2 Graph-based experiments

Calculating incoming links in common

Earlier Jaccard similarity measure served us to select the sample of the most popular red links and create a candidate set. Now we use the particular scores as a feature to find the most probable correspondent page for a red link among the candidates.

We apply Jaccard similarity measure to 2 957 927 pairs of red links and their candidates. Distribution of Jaccard scores for the dataset of pairs is represented in figure 4.2. The maximum number of pairs has the lowest Jaccard score which is about zero. It makes sense as the majority of pairs are false which means that an English candidate article for a red link is not its correspondent article. Then we have normal distribution between scores 0 and 0.6 which is suspicious because it is expected to be some tendencies in extreme positions but not in the middle.

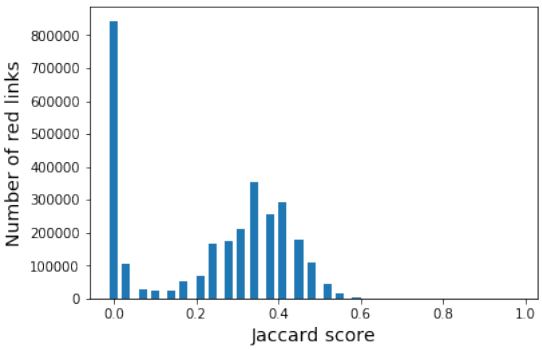

FIGURE 4.2: Incoming links for a red link

Figure 4.3 reveals true impact of Jaccard scores as a similarity model. It shows the fraction of true labels for each score value.

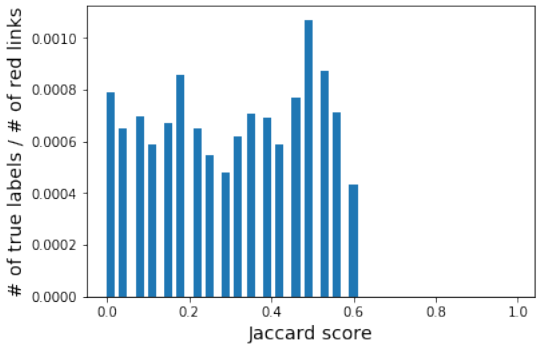

FIGURE 4.3: Incoming links for a red link

Here the distribution is almost uniform which means that true values are similarly distributed among all scores. Moreover the maximum score (0.6 here) where the majority of true values are expected to be found has the smallest number of them. Thus we conclude that the Jaccard score on incoming links seems to be not useful for our task of finding the correspondent articles to red links.

The results for Jaccard similarity metrics were first evaluated through the dataset of unique red links. It gave 49 % in terms of F_1 score.

Next we regarded Jaccard similarity metrics as an independent model with two parameters: a threshold with the best F_1 score and a number of top candidates within which we assume a true candidate is present. We search for this parameters with the train set of 2 337 270 pairs. In figure 4.4 we see the change of F_1 score over all Jaccard scores for the train set.

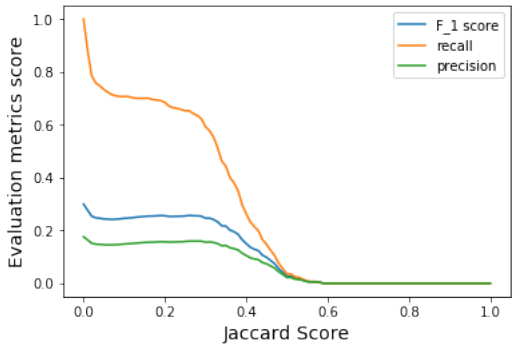

FIGURE 4.4: Evaluation of Jaccard similarity model on incoming links

The model yields the best F_1 score 26 % with a Jaccard score of 0.26 and top 1 candidate. Thus we define 0.26 Jaccard score on incoming links as our threshold for further experiments and set 1 top candidate to choose in the test set. With these parameters, Jaccard similarity model on incoming links on the test set yields F_1 score 23 %. Similar evaluation scores on train and test sets also indicate the reliability of our experiments so far. The top 1 candidate chosen by this model as true is put aside and will be exploited in our multi-factored model.

Calculating concurrent links in common

Another graph characteristics we use is concurrent links for a red link and for candidates. Concurrent links are all the links that occur in the same page as a target

link. We apply Jaccard measure to sets of concurrent links and obtain the following results. Figure 4.5 represents the number of candidate pairs for each score.

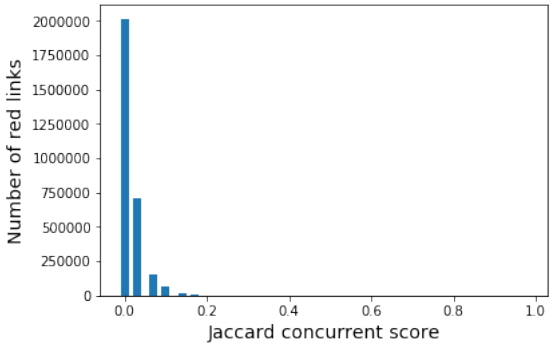

FIGURE 4.5: Concurrent links for a red link

Here the majority of pairs have the lowest score as in the case of Jaccard measure on incoming links. However the fraction of true labels per score value is different (figure 4.6). The majority of true values get the better scores. Significant values are between 0.35 and 0.5 Jaccard scores.

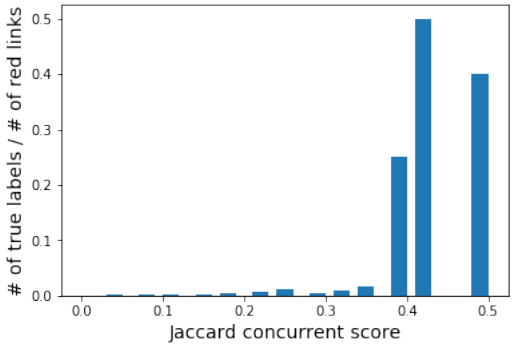

FIGURE 4.6: Concurrent links for a red link

We evaluate Jaccard similarity metrics on concurrent links as an independent model as well. The results of training are shown in the figure 4.7.

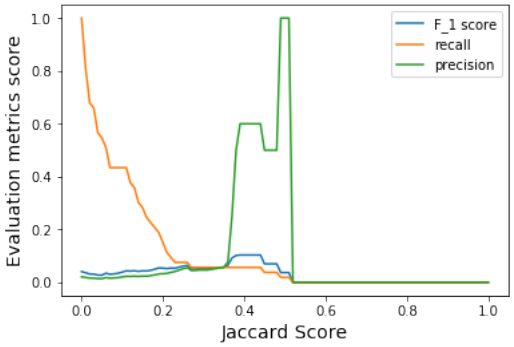

FIGURE 4.7: Evaluation of Jaccard similarity model on concurrent links

After training the model we obtain the best F_1 score 10 % with Jaccard score 0.39 and top 1 candidate. Applying this parameters to the test set yields F_1 score 4 %. The

chosen candidate by Jaccard similarity model on concurrent links is put aside and will be also used to boost our next experiments.

Graph embedding

First, we applied graph embedding techniques to find correspondent pages for English red links[1]. For embedding we assumed that the information we need is all links between articles in Ukrainian Wikipedia and the articles from English Wikipedia for which we are looking the correspondences in Ukrainian Wikipedia. So we excluded all other information about links in English edition as it would add noise in our dataset and make it much bigger. Nevertheless this data was too huge for the library and our hardware so we kept information only on red articles, their 'parent' articles with ids in Ukrainian edition and all the outcoming links of these 'parent' articles in Ukrainian Wikipedia among which could be potential equivalences for red articles. Totally the graph which we embedded had a size of 99 629 nodes and 348 948 edges. Locally Linear Embedding technique did not work for us and it needs further investigation of the cause of fail but SDNE gave us the first embedding of our graph. Yet it was impossible to process it further as the matrix of cosine similarities between nodes was of size 31 GB.

Based on the previous experiments we concluded that this method is not suitable for our task for now. Moreover the graph characteristics we build our graph on is incoming links. And the similarity by incoming links can be calculated without any embeddings which costs much less resources. For that purpose we choose Jaccard similarity metrics which is intersection (common links of red link and a candidate) over union (all links of a red link and a candidate).

4.3.3 Word-based similarity model results

A word-based feature we use to compare red links and candidate articles is their titles. The difficulty here is that titles are of different languages, the majority of red links are in Ukrainian and Cyrillic script, the majority of candidate titles are in English and Latin script. We tackle this obstacle in different ways described below.

Levenshtein edit distance on transliterated titles

Each red link from the sample was transliterated in Latin script and then matched to each of all the candidates that were obtained with the Jaccard similarity on incoming links approach. For matching a Levenshtein edit distance was used. The candidate link which has the lowest edit distance with the red link is considered to be a red link correspondent page in English Wikipedia.

Figure 4.8 shows the distribution of Levenshtein scores among all pairs. The distribution is moved to the left which means that the majority of red links obtain higher Levenshtein scores which is expected as the majority of pairs are false and should get high distance.

[1]Final project for Mining Massive Datasets at UCU, 2018: https://github.com/olekscode/Power2TheWiki

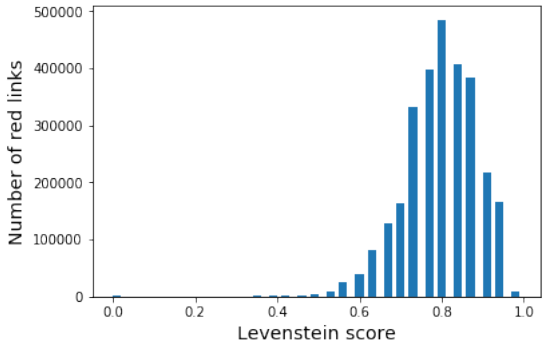

FIGURE 4.8: Levenshtein score on transliterated titles

Next, the fraction of true labels for each score is calculated and we obtain positive results (see figure 4.9). With smaller distance there are more true labels.

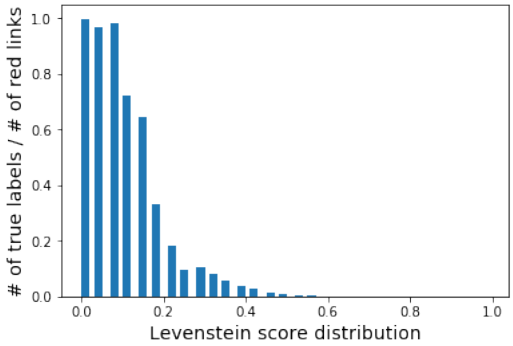

FIGURE 4.9: Fraction of true labels for Levenshtein score on transliterated titles

It means that this similarity measure can be applied for searching correspondent pages for red links.

Application of Levenshtein distance as an independent similarity measure brings us the results shown in the table 4.2. Here recall is 100 % and precision is 47%.

F_1 score 64 %	Precision 47 % Recall 100 %	TP 1501 items FN 0 items FP 1670 items

TABLE 4.2: Levenshtein similarity metrics results

Together they give F_1 score 64% which is the best result obtained for our problem so far. Still we see that we should work on increasing the precision.

When applying Levenshtein edit distance as an independent model on our train set of pairs we get the following results shown in figure 4.10.

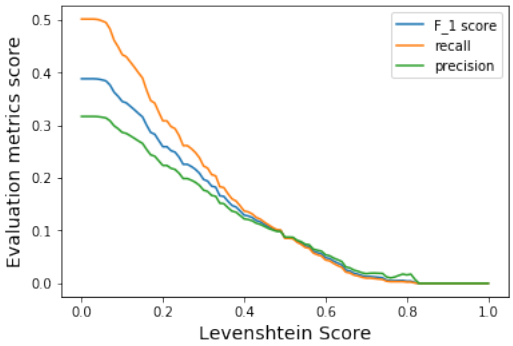

FIGURE 4.10: Evaluation of Levenshtein edit distance similarity model

Here we obtain consistent results as the best F_1 score 39 % goes with the lowest edit distance 0.0. The top n for the best results is also 1 candidate. Applying Levenshtein edit distance model with these parameters to the test set we obtain F_1 score 22 %. The chosen candidate by Levenshtein edit distance model is put aside and will be also used to boost our next experiments.

Cross-lingual embedding similarity model

We applied cross-lingual embedding to a sample of 100 red links. As a result, no correspondent items were found correctly. Nevertheless all suggested items are of the same topical meaning as respective red links. Thus this model does not match correspondent titles but matches topics.

Therefore whereas this model is of no use separately it could be helpful as a part of an ensemble algorithm. We suggest it as our future work.

4.4 Multi-factor Similarity-based Model

Finally, we combined all our similarity metrics as features under the logistic regression model. We appealed to this machine learning algorithm because of two reasons. The first is to make our results on similarity metrics easy to interpret. The second is to improve the results of independent similarity metrics to solve our task.

Since our train and test sets are highly unbalanced we reduced a number of less probable candidates. For that we assume that each similarity metrics choose the most probable candidate among others. Therefore we compose our refined train and test sets from the most probable candidates chosen by each similarity metrics: namely by Jaccard score on incoming links, Jaccard score on concurrent links and Levenshtein edit distance on titles. This way from the train set of 2 337 270 pairs we obtain a train set of 7 608 pairs, from the test set of 620 657 pairs we obtain a set one of 1 905 pairs. In other words, now each Ukrainian red link has three candidates from English Wikipedia which are the three most probable candidates according to the applied similarity metrics before.

As features to train our model we use the scores of three mentioned similarity metrics and also exploit the results of BabelNet.

We used scikit-learn library[2] for training and evaluation of logistic regression model.

[2] https://scikit-learn.org/

The results of applying logistic regression is presented in table 4.3. These are results on the test set.

precision	recall	f_1 score
0.92	0.79	0.85

TABLE 4.3: Logistic regression results (test set)

We obtain F_1 score 85 % which is the best result in comparison to all other considered approaches.

The coefficients of each component are presented in table 4.4

w_0	w_1	w_2	w_3	w_4
3.57	2.18	−0.62	−8.00	1.65

TABLE 4.4: Logistic regression coefficients

Using these coefficients we estimate importance of each feature for the model. This estimation is calculated based on absolute values of the coefficients. The results for the current model are presented in figure 4.11

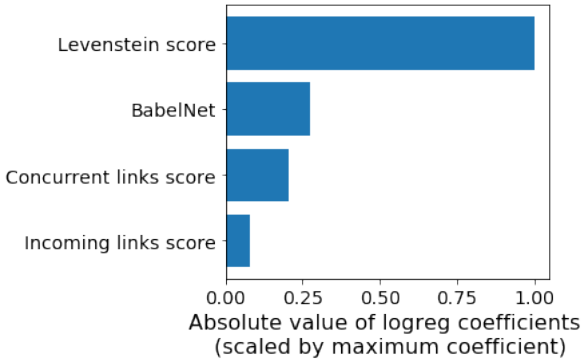

FIGURE 4.11: Relative feature importance for logistic regression model

The highest coefficient corresponds to the most important feature which is Levenshtein edit distance in our case. As we see, word based similarities are the most important in our approach. We hypothesise that the reason for that is that the majority of the red links correspond to nouns, that does not change significantly across languages. This feature is especially boosted given that we are transliterating from Cyrillic to Latin script in order to run this comparison. Future work will be necessary to understand how this metric is affected by pairs of scripts that are more difficult to translate (e.g. from some Asian scripts to English).

Chapter 5

Discussion and Conclusions

In this thesis we presented a solution for resolving red links in the Ukrainian Wikipedia. It is the first and for now the unique prototype for a tool of matching Ukrainian red links with existed articles in the English Wikipedia. All the code is released on github and is open for further use.

We presented a step-by-step Named Entity Linking pipeline, from retrieving data and creating datasets to applying a machine learning model to resolve red links.

The created datasets have been published. The first is a dataset of the most frequent 3 171 Ukrainian red links. They occur in 20 or more articles which have corresponding articles in the English Wikipedia. The second dataset consists of 2 957 927 pairs which are red links and their candidate pages from English Wikipedia. Now the dataset is open for further use and research. We supplied it with a statistics and a thorough analysis on its characteristics which is meant to boost further experiments.

We started with exploring related projects in the Wikimedia community. Found out that no significant previous work had been done to solve this problem, we introduced BabelNet knowledge base as a tool for translating red link items into other languages. As a result we presented its powers for resolving English red links through Ukrainian synsets of BabelNet and vice versa, for resolving Ukrainian red links through English BabelNet synsets.

Next we stated the problem of resolving red links as a Named Entity Linking task. After exploring background work in this field we defined the methodology which is suitable to our task and the main components necessary for the experiments.

We assumed that matching red links with items in other Wikipedia editions could be solved through Wikipedia graph properties and word properties of their titles. Based on that assumption we chose several similarity metrics to find the correspondent page as most similar to a red link by the mentioned properties: Jaccard similarity on incoming links of red links and their candidate pages, Jaccard similarity on links which occur in same pages as red links, Levenshtein edit distance on titles of red links and their candidate pages. Interestingly, we found that a simple metrics such as Levenshtein distance performs better than other more sophisticated approaches. We hypothesise that it happens is due the morphological characteristics of red links. In our thesis we also described the failed experiments such as graph embedding, cross-lingual embedding and red links context retrieval. We provided our considerations on the bottlenecks. Finally we presented results on our multi-factor similarity-based model which combined all previous results of our project. We used logistic regression for a linear model and achieved F_1 score 85 % which is quite good to make a prototype tool for solving this problem.

5.0.1 Limitations and Future Work

Limitations and problems from our approach we divide into three categories: limitations from data, methodology limitations and technical limitations.

First, our dataset is limited to two languages, Ukrainian and English. From one hand, red links that we explore are from the Ukrainian Wikipedia. Red links from other Wikipedia editions could differ thus demanding other approaches to tackle the task. From the other hand, we matched red links only with items from the English Wikipedia. The results could be improved if search the correspondent pages through different Wikipedia editions. The last thing is that all the experiments were conducted on a static data sample from 2018. Red links have the dynamical nature, they change throughout the time, thus new insights could appear if manage red links throughout several data stamps.

As for the methodological limitations, other red links and candidate pages properties could be used as features for our model. For example, the content of Wikipedia pages where red links occur. For now these drawbacks were provoked by resources limitations which is our third point.

With our technical limitations (see Appendix A) we could not apply resources demanding approaches. Thus we chose to reduce data to manageable for our resources sizes.

As a future work we could mitigate mentioned limitations. First, try methods to handle big sizes of the data, for example work on clusters and remote servers. This would permit us to experiment through several languages, add new features and process data throughout the time.

Appendices

Appendix A

Techniques for not running out of memory

During our research we often ran out of memory. In this appendix we describe the problem, technical characteristics of our device and our ways to tackle these obstacles.

Amount of data: 30 GB of raw data.

Device properties:
- 32 GB of RAM
- i7-3930K CPU @ 3.20GHz
- GeForce GTX TITAN, 6 GB of VRAM
- 256 GB of SSD
- Ubuntu 18.04

Solutions to not run out of memory

1. generate samples instead working with a whole dataset;
2. divide samples into files to process them one by one;
3. generate unique ids for items which let split the processes and then combine them;
4. save with data formats that use less memory on a hard disk
    - .npy format. Standard binary file format in NumPy for persisting a single arbitrary NumPy array on disk[1].
    - .pkl format. Python module for data serialization into binary format. It works with different Python data structures such as lists, dictionaries, tuples and it is easy to use. See documentation here[2].

[1] https://numpy.org/devdocs/reference/generated/numpy.lib.format.html
[2] https://docs.python.org/3/library/pickle.html

Bibliography

BabelNet 4.0 and Live Version. URL: https://babelnet.org/.

Bishop, Christopher M (2006). *Pattern recognition and machine learning*. springer.

Bollacker, Kurt et al. (2008). "Freebase: a collaboratively created graph database for structuring human knowledge". In: *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, pp. 1247–1250.

Fabian, MS, K Gjergji, WEIKUM Gerhard, et al. (2007). "Yago: A core of semantic knowledge unifying wordnet and wikipedia". In: *16th International World Wide Web Conference, WWW*, pp. 697–706.

Filling red links with Wikidata, Wikimedia Meta-Wiki. URL: https://meta.wikimedia.org/wiki/Filling_red_links_with_Wikidata.

Final project for the Mining Massive Datasets course at the Ukrainian Catholic University, 2018. URL: https://github.com/olekscode/Power2TheWiki.

Goyal, Palash and Emilio Ferrara (2018). "Graph embedding techniques, applications, and performance: A survey". In: *Knowledge-Based Systems* 151, pp. 78–94.

Jurafsky, Dan and James H. Martin (2019). "Speech and Language Processing". In: *Speech and Language Processing*. Third Edition draft.

Kosub, Sven (2019). "A note on the triangle inequality for the jaccard distance". In: *Pattern Recognition Letters* 120, pp. 36–38.

Lee, John M (2013). "Smooth manifolds". In: *Introduction to Smooth Manifolds*. Springer.

Moro, Andrea and Roberto Navigli (2015). "Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking". In: *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 288–297.

Paranjape, Ashwin et al. (2016). "Improving website hyperlink structure using server logs". In: *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, pp. 615–624.

Parravicini, Alberto et al. (2019). "Fast and Accurate Entity Linking via Graph Embedding". In: *Proceedings of the 2nd Joint International Workshop on Graph Data Management Experiences Systems (GRADES) and Network Data Analytics (NDA)*. GRADES-NDA'19. Amsterdam, Netherlands: Association for Computing Machinery. ISBN: 9781450367899. DOI: 10.1145/3327964.3328499. URL: https://doi.org/10.1145/3327964.3328499.

PetScan tool for Wikimedia. URL: https://petscan.wmflabs.org/.

Ruder, Sebastian, Ivan Vulić, and Anders Søgaard (2017). "A survey of cross-lingual word embedding models". In: *arXiv preprint arXiv:1706.04902*.

Sasaki, Yutaka et al. (2007). "The truth of the F-measure". In: *Teach Tutor mater* 1.5, pp. 1–5.

Shen, Wei, Jianyong Wang, and Jiawei Han (2014). "Entity linking with a knowledge base: Issues, techniques, and solutions". In: *IEEE Transactions on Knowledge and Data Engineering* 27.2, pp. 443–460.

Sherkat, Ehsan and Evangelos E Milios (2017). "Vector embedding of wikipedia concepts and entities". In: *International conference on applications of natural language to information systems*. Springer, pp. 418–428.

Wiki-research-l Digest, Vol 157, Issue 19. URL: https://lists.wikimedia.org/pipermail/wiki-research-l/2018-September/006439.html.

Wikimedia Toolforge for developers. URL: https://tools.wmflabs.org/.

Wikipedia:WikiProject Red Link Recovery/RLRL. URL: https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Red_Link_Recovery/RLRL.

Yujian, Li and Liu Bo (2007). "A normalized Levenshtein distance metric". In: *IEEE transactions on pattern analysis and machine intelligence* 29.6, pp. 1091–1095.

Zheng, Zhicheng et al. (2010). "Learning to link entities with knowledge base". In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 483–491.

Zwicklbauer, Stefan, Christin Seifert, and Michael Granitzer (2016). "Robust and collective entity disambiguation through semantic embeddings". In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, pp. 425–434.