Title (EN): Domain-Specific NER Adaptation

In the past years, the Named Entity Recognition (NER) technology has been under an active development and enjoy a significant increase in popularity and usage in the academic and industrial sphere. Nevertheless, vast majority of the developed NER systems have been developed as general-purpose systems. While they can perform well on multiple domains (macro level), on specific domains (micro level) their performance quality might be low. The ultimate goal of the thesis is to develop domain-specific NER models. Guidelines:

- Get familiar with the NER technology and available NER frameworks.
- Investigate possible datasets for domain-specific training of NER.
- Develop NER training datasets for several selected domains (e.g. sports, politics, music, etc.).
- Train a domain-specific NER model using existing frameworks, such as DBpedia Spotlight or StanfordNER.
- Validate and evaluate the developed domain-specific NER models.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Domain-specific Named Entity Recognition

Bc. Bogoljub Jakovcheski

Department of software engineering
Supervisor: Ing. Milan Dojčinovski

June 17, 2018

Acknowledgements

I would like to thank my family and friends for support during writing this thesis.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46(6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity. However, all persons that makes use of the above license shall be obliged to grant a license at least in the same scope as defined above with respect to each and every work that is created (wholly or in part) based on the Work, by modifying the Work, by combining the Work with another work, by including the Work in a collection of works or by adapting the Work (including translation), and at the same time make available the source code of such work at least in a way and scope that are comparable to the way and scope in which the source code of the Work is made available.

In Prague on June 17, 2018 . . . .

Citation of this thesis

Jakovcheski, Bogoljub. *Domain-specific Named Entity Recognition.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2018.

Abstrakt

Klíčová slova

Abstract

Keywords

Contents

List of Figures

List of Tables

Introduction

Motivation

Named Entity Recognition (NER)[1] is locating and classifying named entities in text into some pre-defined categories such as locations, organizations, person name, sport etc. Today NER is used to different areas from full-text search and filtering to preprocessing tool for other NLP tasks [2].

Most NER applications are trained on a general text and on a specific domain, the problem is that they are optimized for the specific type of data i.e. specific domain. That means that those NER applications can give nice results on texts or domains that are trained, but bad results for texts on a specific domain for which that NER is not trained.

Most of the NER applications are trained on a small number of types. For example, at the moment of writing this thesis, Stanford NER[1] has a model that have maximum 7 types, Dbpedia Spotlight[2] has model with 31 types, spaCy[3] build-in model has 18 types and spaCy Wikipedia scheme model have 4 types.

The main goal of this thesis is to research possibilities of training NER models for a specific domain. To achieve this goal it is necessary to create datasets for certain domains. This research is focused on 3 domains, "POLITICS", "SPORT" and "TRANSPORTATION". Every domain is created with a certain number on types from DBpedia Ontology, then for creating datasets is used DBpedia NIF who gives an opportunity to approaches to information from Wikipedia abstracts, for example, types that annotated words has in those abstracts.

Thesis research which is the quality of trained domains, the impact of the size of the data and the quality of the defined domains.

[1] http://nlp.stanford.edu:8080/ner/
[2] https://www.dbpedia-spotlight.org/demo/
[3] https://spacy.io/usage/linguistic-features

Goals of the thesis

Nevertheless, vast majority of the developed NER systems have been developed as general-purpose systems. While they can perform well on multiple domains (macro level), on specific domains (micro level) their performance quality might be low. The ultimate goal of the thesis is to develop domain-specific NER models. Guidelines:

- Investigate possible datasets for domain-specific training of NER.
- Develop NER training datasets for several selected domains (e.g. sports, politics, music, etc.).
- Train a domain-specific NER model using existing frameworks, such as DBpedia Spotlight or StanfordNER.
- Validate and evaluate the developed domain-specific NER models.

Thesis outline

Chapter 1

Background and related work

1.1 Background

1.1.1 Information extraction

Information extraction first appears in late 1970s within NLP field[4]. Information extraction (IE) [3] is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents. In most of the cases, this activity concerns processing human language texts by means of natural language processing (NLP). Recent activities in multimedia document processing like automatic annotation and content extraction out of images/audio/video could be seen as information extraction.

Another view of that what Information extraction is that automatically building a relational database from information contained in unstructured text. Unlike linear-chain models, general CRFs can capture long distance dependencies between labels [4].

To understand better what IE is let's give trivial example[5]. Imagine receiving an email message with some date in it. So extracting information from mail message and adding to your Calendar is part of IE. Millions of people use this on their daily basis and they are not aware of that how that works and what technology is used for that.

Figure **??** gives us a closer look at what Information extraction (IE) is, and how State-of-the-Art algorithms transform unstructured text to structured sequences understandable for machines.

[4] https://www.slideshare.net/rubenizquierdobevia/information-extraction-45392844 slide 4 of 69

[5] https://ontotext.com/knowledgehub/fundamentals/information-extraction/

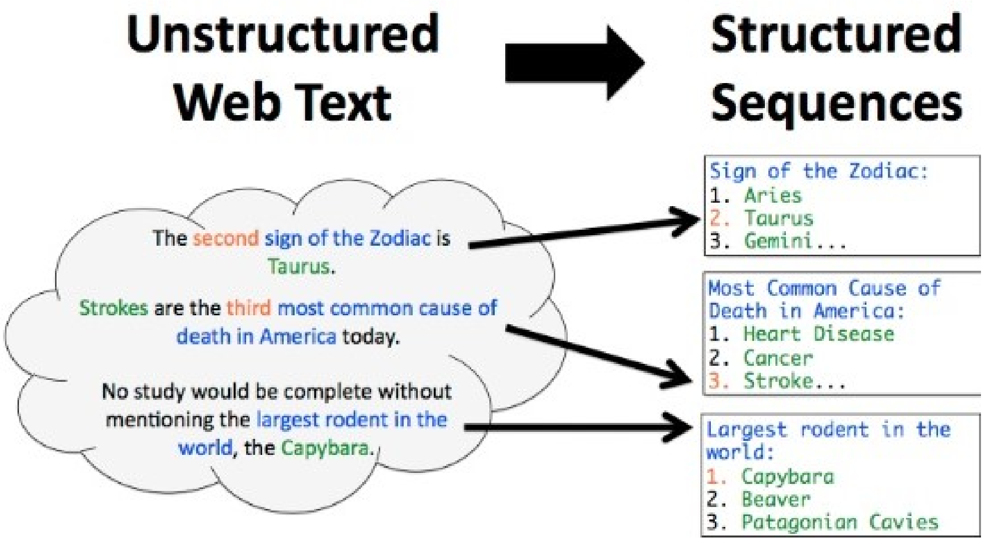


Figure 1.1: Information extraction, downloaded from[6]

1.1.2 Named Entity Recognition

Named Entity Recognition (NER) [5] is the problem of identifying and classifying proper names in text, including locations, such as China; people, such as George Bush; and organizations, such as the United Nations. The named-entity recognition task is, given a sentence, first to segment which words are part of entities, and then to classify each entity by type (person, organization, location, and so on). The challenge of this problem is that many named entities are too rare to appear even in a large training set, and therefore the system must identify them based only on context.

One approach to NER is to classify each word independently as one of either Person, Location, Organization, or Other (meaning not an entity). The problem with this approach is that it assumes that given the input, all of the named entity labels are independent. In fact, the named-entity labels of neighboring words are dependent; for example, while New York is a location, New York Times is an organization.

Most research on NER systems has been structured as taking an unannotated block of text, such as this one:

Jim bought 300 shares of Acme Corp. in 2006.

And producing an annotated block of text that highlights the names of entities:

[Jim]Person bought 300 shares of [Acme Corp.]Organization in [2006]Time.

[6] https://www.slideshare.net/rubenizquierdobevia/information-extraction-45392844

In this example, a person name consisting of one token, a two-token company name and a temporal expression have been detected and classified [1].

Figure 1.2 shows how one NER application can look like. The text in the example is predefined in Stanford NER application and loaded model (Classifier) is also trained by Stanford[7].

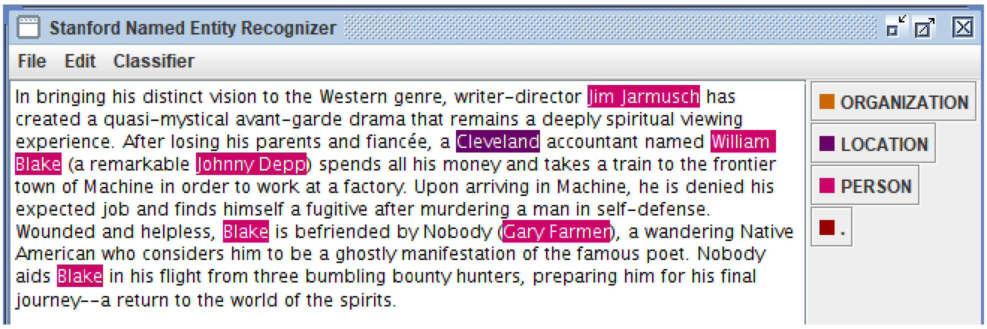


Figure 1.2: Stanford NER GUI with 3 classes model (Location, Person, Organization)

There are several applications or frameworks for NER like Stanford NER, DBpedia Spotlight, spaCy, Chatbot NER, GATE, OpenNLP and so on. Here we will take a look only on the mentioned ones.

1.1.2.1 Stanford NER

Stanford NER[8] is a Java implementation of a Named Entity Recognizer. Named Entity Recognition (NER) labels sequences of words in a text which are the names of things, such as person and company names, or gene and protein names. It comes with well-engineered feature extractors for Named Entity Recognition, and many options for defining feature extractors. Included with the download are good named entity recognizers for English, particularly for the 3 classes (PERSON, ORGANIZATION, LOCATION), and we also make available on this page various other models for different languages and circumstances, including models trained on just the CoNLL 2003 English training data.

Stanford NER is also known as CRFClassifier. The software provides a general implementation of (arbitrary order) linear chain Conditional Random Field (CRF) sequence models. That is, by training your own models on labeled data, you can actually use this code to build sequence models for NER or any other task [6].

[7] https://nlp.stanford.edu/software/CRF-NER.html#Models
[8] https://nlp.stanford.edu/software/CRF-NER.html

1.1.2.2 DBpedia Spotlight

DBpedia Spotlight[9] [7] is a tool for annotating mentions of DBpedia resources in text. This allows linking unstructured information sources to the Linked Open Data cloud through DBpedia. DBpedia Spotlight performs named entity extraction, including entity detection and name resolution (in other words, disambiguation). It can also be used for named entity recognition, and other information extraction tasks. DBpedia Spotlight aims to be customizable for many use cases. Instead of focusing on a few entity types, the project strives to support the annotation of all 3.5 million entities and concepts from more than 320 classes in DBpedia. The project started in June 2010 at the Web Based Systems Group at the Free University of Berlin.

1.1.2.3 spaCy

spaCy[10] [8] is an open-source software library for advanced Natural Language Processing, written in the programming languages Python and Cython. It offers the fastest syntactic parser in the world. The library is published under the MIT license and currently offers statistical neural network models for English, German, Spanish, Portuguese, French, Italian, Dutch and multi-language NER, as well as tokenization for various other languages.

1.1.2.4 GATE

General Architecture for Text Engineering or GATE[11] [9] is a Java suite of tools originally developed at the University of Sheffield beginning in 1995 and now used worldwide by a wide community of scientists, companies, teachers and students for many natural language processing tasks, including information extraction in many languages.

GATE includes an information extraction system called ANNIE (A Nearly-New Information Extraction System)[12] which is a set of modules comprising a tokenizer, a gazetteer, a sentence splitter, a part of speech tagger, a named entities transducer and a coreference tagger. ANNIE can be used as-is to provide basic information extraction functionality, or provide a starting point for more specific tasks.

1.1.2.5 OpenNLP

The Apache OpenNLP library[13] is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP

[9] https://www.dbpedia-spotlight.org/

[10] https://spacy.io/

[11] https://gate.ac.uk/

[12] http://services.gate.ac.uk/annie/

[13] http://opennlp.apache.org/docs/1.8.4/manual/opennlp.html#intro.description

tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution. These tasks are usually required to build more advanced text processing services. OpenNLP also included maximum entropy and perceptron based machine learning.

The goal of the OpenNLP project will be to create a mature toolkit for the abovementioned tasks. An additional goal is to provide a large number of pre-built models for a variety of languages, as well as the annotated text resources that those models are derived from.

1.1.2.6 Chatbot NER

Chatbot NER[14] is heuristic based that uses several NLP techniques to extract necessary entities from chat interface. In Chatbot, there are several entities that need to be identified and each entity has to be distinguished based on its type as a different entity has different detection logic.

1.1.3 RDF/NIF

The Resource Description Framework (RDF)[10] is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model. It is a framework for describing resources on the web; it is designed to be read and understood by computers.

The information in RDF is represented by subject-predicate-object, known as triples. Triples are written in one of RDF notations: RDF/XML, RDFa, N-Triples, Turtle, JSON-LD and stored in a triplestore [11].

RDF [12] has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed.

Natural Language Processing Interchange Format (NIF)[15] [13] is an RDF-based format. The classes to represent linguistic data are defined in the NIF Core Ontology. All ontology classes are derived from the main class nif:String which respresents strings of Unicode characters.

1.1.4 DBpedia

DBpedia [14] is a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects. This structured information resembles an open knowledge graph (OKG) which is available for everyone on the Web. A knowledge graph is a special kind of database which stores knowledge in a machine-readable form and provides a means for

[14]https://haptik.ai/tech/open-sourcing-chatbot-ner/
[15]http://aksw.org/Projects/NIF.html

information to be collected, organised, shared, searched and utilised. Google uses a similar approach to create those knowledge cards during search.

DBpedia data is served as Linked Data, which is revolutionizing the way applications interact with the Web. One can navigate this Web of facts with standard Web browsers, automated crawlers or pose complex queries with SQL-like query languages (e.g. SPARQL).

At the time of writing this thesis the last version of DBpedia is 3.7.

1.1.4.1 DBpedia NIF

DBpedia [15] currently primarily focus on representing factual knowledge as contained in Wikipedia infoboxes. A vast amount of information, however, is contained in the unstructured Wikipedia article texts. In order to broaden and deepen the amount of structured DBpedia data, we are going a step further.

With the representation of wiki pages in the NLP Interchange Format (NIF) we provide all information directly extractable from the HTML source code divided into three datasets:

- nif-context: the full text of a page as context (including begin and end index)
- nif-page-structure: the structure of the page in sections and paragraphs (titles, subsections etc.)
- nif-text-links: all in-text links to other DBpedia resources as well as external references

These datasets will serve as the groundwork for further NLP fact extraction tasks to enrich the gathered knowledge of DBpedia.

For the purposes of this thesis we will use DBpedia NIF dataset version 2016-04 (dbpv=2016-04).

1.1.4.2 DBpedia ontology

The DBpedia Ontology is a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia. The ontology currently covers 685 classes which form a subsumption hierarchy and are described by 2,795 different properties.

Since the DBpedia 3.7 release, the ontology is a directed-acyclic graph, not a tree. Classes may have multiple superclasses, which was important for the mappings to schema.org. A taxonomy can still be constructed by ignoring all superclasses except the one that is specified first in the list and is considered the most important [16].

Dbpedia ontology classes can be found here [16]

[16] `http://mappings.dbpedia.org/server/ontology/classes/`

The DBpedia Ontology currently contains about 4,233,000 instances. Figure 1.3 shows the number of instances for several classes within the ontology. [http://wiki.dbpedia.org/services-resources/ontology]

Class	Instances
Resource (overall)	4,233,000
Place	735,000
Person	1,450,000
Work	411,000
Species	251,000
Organisation	241,000

Figure 1.3: Dbpedia Ontology - Instances per class

1.1.5 Apache Jena

Apache Jena[17] [17] is an open source Semantic Web framework for Java. It provides an API to extract data from and write to RDF graphs. The graphs are represented as an abstract "model". A model can be sourced with data from files, databases, URLs or a combination of these. A Model can also be queried through SPARQL 1.1.

1.1.6 SPARQL

SPARQL [11] is an RDF query language, that is, a semantic query language for databases, able to retrieve and manipulate data stored in Resource Description Framework (RDF) format. SPARQL works for any data source that can be mapped to RDF.

SPARQL allows users to write queries against key-value data or, more specifically, data that can be mapped to RDF. The entire database is thus a set of subject-predicate-object triples.

The SPARQL standard [18] is designed and endorsed by the W3C and helps users and developers focus on what they would like to know instead of how a database is organized.

[17] `https://jena.apache.org/index.html`
[18] `https://ontotext.com/knowledgehub/fundamentals/what-is-sparql/`

In Listing 1.1 is an example of SPARQL query where we are selecting 10 abstracts from DBpedia NIF who has ontology type PoliticalParty and their PageRank and sort descending by PageRank.

Listing 1.1: SPARQL example

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbo:<http://dbpedia.org/ontology/>
PREFIX vrank:<http://purl.org/voc/vrank#>

SELECT DISTINCT ?s ?v
FROM <http://dbpedia.org>
FROM <http://people.aifb.kit.edu/ath/#DBpedia_PageRank>
WHERE{
        ?s rdf:type dbo:PoliticalParty .
        ?s vrank:hasRank/vrank:rankValue ?v.
}
ORDER BY DESC(?v) LIMIT 10
```

1.2 Related work

In this section we will compare our approach and our chosen domains.

1.2.1 Domain specific Named Entity Recognition

Traditionally Named Entity Recognition (NER)[18] systems have been built using available annotated datasets (like CoNLL, MUC) and demonstrate excellent performance. However, these models fail to generalize onto other domains like Sports and Finance where conventions and language use can differ significantly. Furthermore, several domains do not have large amounts of annotated labeled data for training robust Named Entity Recognition models. With specifying the domain we can create a bigger model with more annotated words and reading the whole text will be same or even faster that reading text with a global domain.

Chapter 2

Domain specific named entity recognition

In this chapter we will go through the whole process of transforming raw DBpedia datasets to datasets that are ready for training a model with Stanford NER and how to train a model with Stanford NER. Section 2.1 explains the process of cleaning the data from DBpedia NIF datasets and preparing them for processing. In Section 2.2 we explain how we choose "POLITICS", "SPORT" and "TRANSPORTATION" domains. Section 2.3 shows all ontology types that we retrieve for every domain and grouping them to more specific ontology type. In Section 2.4 is explained the process of preparing datasets for training in Stanford NER. And finally in Section 2.5 is shown how to train a datasets with Stanford NER.

2.1 Data pre-processing

To be able to create domain specific datasets ideally we need some big raw data. We choose data from DBpedia NIF Datasets (for more information about DBpedia NIF see Section 1.1.4.1) for the English language in .ttl format. From here we needed only 2 datasets, and that nif-context (or nif-abstract-context) and nif-text-links.

Another dataset that we needed was DBpedia instance types dataset, found at DBpedia download page[19] also in .ttl format. This dataset contains all types of nif-text-links that occurrence at nif-abstract-context file.

So how all this dataset are connected between themselves? Let say that we have abstract for Alexander the Great. In nif-text-links file we have all words from the abstract that has annotation, but we still don't know their type. So here comes instance types file where based on link from nif-text-link (eg.http://dbpedia.org/resource/Philip_II_of_Macedon) we can find the type

[19] http://wiki.dbpedia.org/downloads-2016-04

of annotated word (word Philip II has ontology type Monarch), but of course, there can be a case that some words cannot be found on instance types file and automatically have no type, or in our case ontology type O (O stands for OTHER).

But now, let us explain deeply how we process and clean data from the datasets. First, we define small test dataset to check how fast we can process data. Running that dataset on downloaded files without any cleaning on data takes too long. So we said that converting the datasets from RDF format to binary format (.ttl to .hdt) with RDF/HDT tool[20] will be faster. HDT (Header, Dictionary, Triples)[19] is a compact data structure and binary serialization format for RDF that keeps big datasets compressed to save space while maintaining search and browse operations without prior decompression. So we converted the datasets and reran the algorithm again. There were some improvements, but not satisfying for our purposes. Our next solution was to clean datasets from unused data for our aims. The final result after cleaning was a smaller datasets, for instance, nif-abstract-context file from 7.78GB now has 2.99GB, another big improvement was nif-text-links file who is reduced to 10.5GB from 44.6GB and at the end we also clean instance-types file, but here we don't record any mayor memory improvements. Again we rerun the algorithm, of course, there were improvements, but as well as previous the time that algorithm runs, was not acceptable for us. To give an illustration, the time needed to find all types from one abstract in a worst case, to read nif-text-links and instance type files until the end was around 3.5 minutes. Therefore once more we converted our cleaned datasets from RDF format(.ttl) to binary format(.hdt). And how in previous running there were again improvements, but those improvements don't fulfill our expectations. The final thing that we have to save us was creating a dataset tree only for nif-text-links and instance-types files. For nif-text-links file we created a tree where we have folders from "a-z", also special characters folders and other folder(this folder contains data that have a lower occurrence, let say & character or letters that are not part of the English alphabet) and folders from "a-z" has subfolders also from "a-z".

To give a closer look how we create that tree, let say that we have an abstract for Volkswagen Golf MK3, so the link for that abstract would be http://dbpedia.org/resource/Volkswagen_Golf_Mk3 and this link will be stored to "v" folder and "o" subfolder, because the title of the abstract is Volkswagen Golf MK3, where we need only first 2 letters from the first word, in this case word Volkswagen. With this, we have a smaller dataset where we can read the whole one very fast.

For instance types file we modified the algorithm for creating a data tree. Here because of lower range data we have created only files from "a-z", of course, special characters files and other file.

Finally, we rerun the algorithm, and the time to process one abstract, at

[20] http://www.rdfhdt.org/

worst case, takes no longer than 1 minute. Now we were ready to take next steps to retrieve types (see Section 2.3), create domains (see Section 2.2) and prepare data for Stanford NER (see Section 2.4).

2.2 Domain specification

As we said earlier most of the NER application are trained on same domains, like ”PERSON”, ”ORGANIZATION” and ”LOCATION”. These 3 domains are widely spread all over the applications and perform nice results on text from this domains. So what we need is something that is not already trained or there is a small usage of that domain. After some research, we find out that ”TRANSPORTATION” domain is not a popular domain for NER applications, respectively in time of writing the thesis we don’t find any usage of this specific domain. So there is the possibility to create this specific domain. Types that we retrieve for this domain and groping them to more specific types are more deeply explained in Types retrieval (see Section 2.3). We have our first domain, but at least 2 more domains are needed to be able to make some experiments and conclusion.

Ideally will be those domains to have some connection between them and again not to be already widespread. So we look at ontology types that are retrieved for ”TRANSPORTATION” domain, and there are types like Airport, Bridge, MetroStation and so on. This indicates to us that next domain can be ”POLITICS”. Why? Because some airports, bridges or metro stations bear names of Politicians. For instance airport in Prague, Czech Republic is named by the last president of Czechoslovakia, Vaclav Havel. Or another example is that some bridges in the United States are named by famous politicians, like Presidents. The types that contains this domain are explained in Section 2.3. The second domain is chosen, so we need at least one more domain to keep up with other NER applications.

With the requirements that we set for choosing a domain, which domain to choose was not easy at all. After a big research, also referring to ontology types from previous two domains and some NER applications (see Section 1.1.2) we find an opportunity to create the last ”SPORT” domain. Now we should check on DBpedia ontology classes page (see Section 1.1.4.2) how many ontology types we have for this domain. At the time of writing this thesis there were around 170 ontology types, which is very good number for creating a domain (for more see Section 2.3).

After we complete choosing of domains, the next big think was to choose the right ontology types for every specific domain and if it is needed or make sense group those types to more specific type. This is totally covered and explained at Section 2.3.

2.3 Types retrieval

After we solved the problem of that how effectively run the algorithm to find all types from the abstract and choose domains, next issue was which types we want to be part of our domains and also which types we want to retrieve from Dbpedia. Worth mentioning that we will use the same ontology types for retrieving the abstracts links from Dbpedia and creating a domain models. For example the type "Politician" will be used to retrieve links from Dbpedia that has that type, and also "Politician" type will be use to annotated words, for instance Barack Obama will have type of "Politician" (we will give more details on section 2.4).

In DBpedia ontology classes page[21] we can see all types that DBpedia ontology has. Those ontology types are the same in instance types file also. Now we are facing with the fact that if we choose very small group of ontology types, at the experiment point we will have minor range of annotated words and experiments won't be relevant. On the other hand, if we go too deep to ontology types, we will have a lot of annotated words, which on one hand is good, but training the model will take a lot of time and memory, and there is a possibility that we will reach memory exception, or because of big group of types training will never end.

After some testing with the number of retrieved types we finally found the best selection of types, in total we choose 283 ontology types for all domains.

Now let us explain more deeply every single domain and which types has that domain. We have 3 domains (see Section 2.2 for that how we choose those domains) "POLITICS", "SPORT" and "TRANSPORTATION".

In "POLITICS" domain we retrieve in total 26 types, found at Appendix A.1, which we sort in 11 more specific types like Ambassador, Chancellor, Congressman, Deputy, Governor, Lieutenant, Mayor, MemberOfParliament, Minister, President, PrimeMinister, Senator, VicePresident and VicePrimeMinister are joined together in one specific domain Politician, other types we leaved as it is, because if we group them the types wouldn't give any sense.

We do the same for "SPORT" domain where we retrieve in total 171 types, found in Appendix A.2, so those types, same as "POLITICS" domain, are more specified in 8 types, like SportClub, SportsLeague, SportsTeam, Athlete, Coach, OrganizationMember, SportsManager and SportsEvent. Grouping of types is also shown in appendix A.2. This domain is a nice example of that even we retrieve quite a big number of types, we can reduce that number with more specific types which further don't lose the sense of type. For instance "David de Gea" has a type of SoccerPlayer, but after processing will have type of Athlete, which gives sense, because any type of sport player is an athlete.

At the end we repeat the process for "TRANSPORTATION" domain, where we retrieve in total 86 types. Retrieved types can be found in Ap-

[21] `http://mappings.dbpedia.org/server/ontology/classes/`

pendix A.3. Those types are after minimized in 14 more specific types like Aircraft, Automobile, Locomotive, MilitaryVehicle, Motorcycle, On-SiteTransportation, Rocket, Ship, SpaceShuttle, SpaceStation, Spacecraft, Train, PublicTransitSystem and Infrastructure. The logic of that who we create more specific ontology types is same as in ”POLITICS” or ”SPORT” domain.

The reason why we group ontology types to more specific ones is that, that when the dataset has a smaller number of types, training a model with Stenford NER is more faster and requires less memory for training. Another reason is faster providing a NER, because is needed to read less types and also the overall results after testing with same data perform better than when ontology types where not grouped.

2.4 Data transformation

We define domains as well their types that we will retrieve and process, now we should put everything together and prepare data for Stanford NER application. In Data pre-processing (see Section 2.1) we explain how we handled the data downloaded from web and we briefly touch how those data will be prepared for training in Stanford NER application.

The final thing that is missing is how we will choose which abstracts will be part of our models. Because our goal is to create models with different number of abstracts we need some strict order of retrieved links from DBpedia dataset. The solution that we choose who fits to our requirements is PageRank. PageRank [20] is an algorithm used by Google Search to rank websites in their search engine results. So with a prepared and tested SPARQL queries on www.dbpedia.com/sparql and with help of Apache Jena framework (see Section 1.1.5) we implemented retrieving links, on Java, on DBpedia endpoint[22]. After retrieving those data, based on their PageRank we search does retrieved link is part on our abstract file. If link is found in nif-abstract dataset it's written to two files, one file is where are written all abstracts from every domain and another file is file for that specific domain. Those files are creating in RDF format, with n-triples, that means that there is subject, in our case that is the link of abstract, then predicate who has isString annotation which tells that next triple contains the abstract text and finally object where abstract text is placed. Next thing that we need to do is to find all annotated words from abstract and their types. The algorithm of finding types is explained in Section 2.1. What is not mention there is that after finding the types, the abstract is written to file, where on first position is word and on the second position is the type of that word, if there is any, if not the type is O. Final step is to prepare data to be able to train models in Stanford NER with the types that we define in Section 2.3. Because files contains all types that were found on the abstracts we need to clean and group them, as well to create a

[22] `http://www.dbpedia.com/sparql`

coarse and fine grained files. The algorithm is very simple, it reads the files who already has all types and if type is part of our retrieved types then either type is leaved as it is, or is grouped to more specific type, for instance if word has type Ambassador, then after filtering that word will have Politician type. The same is for coarse grained annotation, but here proper types after filtering are ”POLITICS”, ”SPORT” or ”TRANSPORTATION” type. The whole process is also illustrated at Algorithm 1. Interesting fact is that, that when we retrieve links from DBpedia with a specific ontology types types, some links there has types that are not even part of our domain. Here are some interesting links that we catch:

- http://dbpedia.org/page/Orbital_period
- http://dbpedia.org/page/Pregnancy
- http://dbpedia.org/page/Melody
- http://dbpedia.org/page/ITunes
- http://dbpedia.org/page/Tachycardia
- http://dbpedia.org/page/Shortwave_radio
- http://dbpedia.org/resource/UTC-05:00

Retrieve links from DBpedia NIF Dataset based on their PageRank;
if *Retrieved link is found at nif-abstract dataset* **then**
    write value from nif-abstract dataset to file
else
    go to next retrieved link and repeat steps
end
Read new file with values from nif-context and get abstract links;
Check does that link is consists in nif-text-links dataset;
if *link consists in nif-text-links* **then**
    Get all values (links) from nif-text-links dataset;
    Search for ontology types in instance-types dataset;
    if *Link from nif-text-links exists in instance-types* **then**
        Parse value and return ontology type;
    else
    end
    Write abstract text to domain specific file with founded type of the word, as only word and the type at a line;
else
    Write abstract text to domain specific file with word and O type at a line;
end
Read created domain specific files and clean unnecessary types;
if *Type equals some of retrieved types* **then**
    Leave type as it is or group type and write to two domain specific files in coarse and fine grained;
else
    Rewrite the type to "O" and write to two domain specific files in coarse and fine grained;
end
Write to two domain specific files in coarse and fine grained;

Algorithm 1: Algorithm for preparing datasets ready for training in Stanford NER

2.5 Model generation

With the created files from Section 2.4 now we can start training models. At Stanford NER CRF FAQ webpage[23] is a very nice explanation of that how to train own model with Stanford NER. We follow those steps and used pretty much the same NER properties file with a small correction where we had to add 2 more flags to be able to train big models. Those two flags are saveFeatureIndexToDisk=true, which is used on every properties file and for creating a models in fine grained we use useObservedSequencesOnly=true.Flag saveFeatureIndexToDisk stands for saving the feature name's to disk that aren't actually needed while the core model estimation (optimization) code is run. More interesting is useObservedSequencesOnly flag. It's stands for labeling only adjacent words with label sequences that were seen next to each other in the training data. For some kinds of data this actually gives better accuracy, for other kinds it is worse. After testing on a small model with only 40 abstracts and model with 300 abstracts we find out that for creating a fine grained model with 40 and more abstract this flag gives us better results, while on coarse grained models this flag gives worst results, the exception are models with 500 abstracts where we should use this flag to reduce memory usage. The whole properties file with all used flags can be found in Appendix B.

After creating a properties files, training models is very easy with only one command, where unlike command from Stanford we add Xmx Java option, because standard command use only 4GB of RAM, which for our purposes is not enough for training big models.

Command for training model ran from the stanford-ner folder:

```
java -Xmx11g -cp stanford-ner.jar
edu.stanford.nlp.ie.crf.CRFClassifier -prop
locationAndnameOfPropFile.prop
```

2.5.1 Training datasets

For the aim of our experiments we have trained 57 models. As mentioned earlier for training we have used Stanford NER application explained in Section 1.1.2.1. We have two types of models, coarse-grained and fine-grained, also those model types are divided in to "POLITICS", "SPORT" or "TRANSPORTATION" specific domains and a global domain who contains all abstracts from every domain. To give an illustration, for dataset with 100 retrieved abstract we will have 4 coarse-grained models (global domain and 3 specific domains), and similarly for a fine-grained models, so in total we have 8 trained models for every dataset. We created 7 different datasets with 10 abstracts, 20 abstracts, 40 abstracts, 100 abstracts, 300 abstracts, 400 abstracts and 500 abstracts. Each of this datasets has 8 trained models and

[23] `https://nlp.stanford.edu/software/crf-faq.html`

we have one dataset that have also 500 abstracts, but those abstracts are not the same like the previous dataset. This dataset contains abstracts that have lower PageRank value and has only one trained model with abstracts from every domain in fine grained.

Chapter 3

Experiments

There are parameters of the computer used for tests shown in Table 3.1.

Table 3.1: Testing computer parameters

Part	Description
CPU	2.00 GHz Intel(R) Core(TM) i5-4310U
MEM	16 GB DDR3L
OS	x86_64 Windows 10 Pro
DISK	240GB SSD Kingston

We have provide various types of experiments. In next sections we will discuss more about every provided experiment. The order of the abstracts is based on PageRank as explained in section 2.4.

3.1 Goals of the experiments

We set a few goals of the experiments. First of all we waned to test does we will get better results if we run the model of all domains in coarse grained, against the model of all domains in fine grained. In this test we run the models also with all domains texts. Then we get those models and we run it with specific domain texts, in both fine and coarse grained. Also we make experiments with specific domain model run with domain specific texts, for example, politics domain model in coarse grained is run with politics domain text also annotated in coarse grained, politics domain model in fine grained is run with politics domain text also annotated in fine grained, and the same for sport and transportation domains.

3.2 Evaluation metrics

The success of NER systems is exposed to F_1 score (F-score or F-measure). F_1 [21] score is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score: p is the number of correct positive results divided by the number of all positive results returned by the classifier, and r is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive). The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. Written in formula, the $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$.

3.3 List of experiments

With our trained models we made a few experiments. First one is the model that has 300 abstract on every domain(900 abstract in total). This is our main model and other experiments that we will provide like models that has lower or higher number of abstracts or experiments where model has more abstracts that a test file or vice-verse, all those results will be compared with the results obtained from main experiment.

3.3.1 Main experiment

This is our main experiment where other experiments will be compared with this one. This model is trained with top 300 Wikipedia abstracts for every domain. Algorithm for preparing the data for training model explained in section 2.4 takes 8622805705290 nanoseconds or 2.40 hours. The model is trained in coarse grained and takes 844.63 seconds in optimization and 873.7 seconds on CRFClassifier training.

3.3.1.1 Global domain models

First experiment that we do with this model is that we run it with the same text that model is trained in coarse grained. Results are not bad at all, we are above 95% as shown in Table 3.2, which is great number for such middle weight model. With such results, someone will say that those are nice results and other experiments will only have a worst results. But let see how model will behaves when we tested with abstracts for every specific domain.

Entity	P	R	F1
POLITICS	0,9872	0,9462	0,9662
SPORT	0,9846	0,9629	0,9736
TRANSPORTATION	0,9940	0,9823	0,9881
Totals	**0,9875**	**0,9625**	**0,9748**

Table 3.2: Outcomes of base experiment run to be used as reference for subsequential experiments

Table 3.3 shows the output of model when is tested with abstracts from a ”POLITICS” domain. As we said in Section 2.4 this type of abstract has the biggest word annotation. Result is not even close with the result from previous experiment. Also, trained model annotated words with a ”TRANSPORTATION” domain, where the test file don’t have any word with that annotation.

Entity	P	R	F1
POLITICS	0,9839	0,4025	0,5713
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9792**	**0,4025**	**0,5705**

Table 3.3: Outcomes of base model in coarse grained run with ”POLITICS” abstracts

Table 3.4 gives us results from abstracts from ”SPORT” domain. Here we have the same results like in first experiment, but because trained model annotated some words with a ”POLITICS” or ”TRANSPORTATION”, even those that our test file contains only abstracts from ”SPORT” domains and words has only ”SPORT” type, the overall result is only a little bit lower that the first experiment.

Entity	P	R	F1
POLITICS	0,0000	1,0000	0,0000
SPORT	0,9846	0,9628	0,9736
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9819**	**0,9628**	**0,9722**

Table 3.4: Outcomes of base model in coarse grained run with ”SPORT” abstracts

Table 3.5 provide outcome with testing with abstracts only from ”TRANSPORTATION” domain. As in the previous experiment, the result now is almost the same like in first experiment, but even thought that trained model, as in previous 2 experiments, annotated words with a ”SPORT” type, the overall results is better that the experiment where test file contains all abstracts from every domain.

Entity	P	R	F1
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	0,9940	0,9822	0,9880
Totals	**0,9861**	**0,9822**	**0,9841**

Table 3.5: Outcomes of base model in coarse grained run with ”TRANSPORTATION” abstracts

In conclusion with this kind of experiments we can say that it is not a good idea to train a model with all chosen domains and then use texts from specific domain to perform NER.

After we finish the experiments with model that is trained with all abstracts from every domain in coarse grained, we waned to see the impact of model that is trained with same abstracts, but now annotated in fine grained. To train this model we needed 3250.9 seconds from which 3207.45 seconds for optimization. Table 3.6 shows the results of provided experiment where we can see that we have a little bit more better total result than experiment in Table 3.2.

Entity	P	R	F1
Aircraft	1,0000	1,0000	1,0000
Athlete	1,0000	0,9802	0,9900
Automobile	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9820	0,9909
PoliticalParty	0,9860	0,9628	0,9743
Politician	1,0000	0,9353	0,9665
PublicTransitSystem	0,9919	0,9839	0,9879
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,9796	0,9683	0,9739
SportsEvent	1,0000	0,9242	0,9606
SportsLeague	0,9647	0,9805	0,9725
SportsManager	1,0000	0,9423	0,9703
SportsTeam	1,0000	0,9805	0,9902
Train	1,0000	1,0000	1,0000
Totals	**0,9880**	**0,9712**	**0,9795**

Table 3.6: Outcomes of base experiment in fine grained run to be used as reference for subsequential experiments

Then we tested our model with abstracts from "POLITICS" domain. How we can see from Table 3.7 there is some improvements on overall result unlike the experiment in coarse grained, but no satisfying at all. As well table shows that some words again are annotated with types from "SPORT" and "TRANSPORTATION" domain.

Entity	**P**	**R**	**F1**
Election	0,0000	0,0000	0,0000
PoliticalParty	0,9860	0,9628	0,9743
Politician	1,0000	0,1849	0,3120
PublicTransitSystem	0,0000	1,0000	0,0000
Ship	0,0000	1,0000	0,0000
SportsLeague	0,0000	1,0000	0,0000
Totals	**0,9825**	**0,4072**	**0,5758**

Table 3.7: Outcomes of base model in fine grained run with "POLITICS" abstracts

After that we rerun the experiment, but now with abstracts from "SPORT" domain. In Table 3.8 we can see minor growth of the results unlike experiment in Table 3.4, but this improvements are so small that are almost unimportant. Also our model annotated some words with types from "POLITICS" and "TRANSPORTATION" domain which the test file don't have those types at all.

Entity	**P**	**R**	**F1**
Athlete	1,0000	0,9802	0,9900
Coach	1,0000	1,0000	1,0000
Politician	0,0000	1,0000	0,0000
SportsClub	0,9794	0,9680	0,9737
SportsEvent	1,0000	0,9242	0,9606
SportsLeague	0,9678	0,9805	0,9741
SportsManager	1,0000	0,9423	0,9703
SportsTeam	1,0000	0,9804	0,9901
Train	0,0000	1,0000	0,0000
Totals	**0,9821**	**0,9716**	**0,9768**

Table 3.8: Outcomes of base model in fine grained run with "SPORT" abstracts

Finally the last experiment with this model are the abstracts from "TRANSPORTATION" domain. Table 3.9 shows the output of the provided experiment, where like in previous 2 experiments we can notice a very little improvements on results, from experiment in Table 3.5, who again can be unimportant.

As in previous experiments similarly here model annotated some words with types from other 2 domains, which test file does not even contains.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Automobile	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9820	0,9909
Politician	0,0000	1,0000	0,0000
PublicTransitSystem	0,9918	0,9837	0,9878
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,0000	1,0000	0,0000
SportsTeam	0,0000	1,0000	0,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9862**	**0,9881**	**0,9871**

Table 3.9: Outcomes of base model in fine grained run with ”TRANSPORTATION” abstracts

Provided experiments with the model who is trained with all abstracts from every domain annotated in fine grained, overall provide a very little improvement on results on every experiment. With that observation trained model annotated in fine grained is better to use instead of the model that is annotated in coarse grained. Another benefit of this type of model is that we can see which types are annotated and their results. But, because those improvements are small and is needed almost four times more time to train a fine grained model, maybe the better solution will be a models trained in coarse grained, everything depends on us. Does we want to trained models faster or we want to be more precise.

3.3.1.2 Evaluation of domain specific models

After completing experiments with a global domains in coarse and fine grained, now we will make experiments with models for specific domains.

To train ”POLITICS” domain specific model we need 66.7 seconds in total from which 59.53 seconds spend on optimization. In Table 3.10 the experiment is provided with model trained only with abstracts from ”POLITICS” domain and run with the same texts that model in trained, in coarse grained. The result here is better than experiment in Table 3.3, but worse that experiment provided with global domain in Table 3.2. This can be cause by the fact that model has biggest number of annotated words.

Entity	P	R	F1
POLITICS	0,8039	0,6779	0,7355
Totals	**0,8039**	**0,6779**	**0,7355**

Table 3.10: Outcomes of ”POLITICS” base model in coarse grained run with ”POLITICS” abstracts

We repeat previous experiment, but now everything in fine grained. Time for training this kind of model i total was 163.5 seconds, from which 155.94 seconds spend on optimization. Table 3.11 shows that this kind of model provides better result that coarse grained model and the experiment provided in Table 3.7, but again worst than model trained with all abstracts (see Table 3.6).

Entity	P	R	F1
Election	0,8240	0,6398	0,7203
PoliticalParty	0,8100	0,7006	0,7513
Politician	0,8599	0,7234	0,7858
Totals	**0,8354**	**0,6980**	**0,7606**

Table 3.11: Outcomes of ”POLITICS” base model in fine grained run with ”POLITICS” abstracts

In conclusion with provided 2 experiments and from this point of view, for this domain we can say that training a specific model will give better results and will perform faster that global domain tested with text from specific domain. On the other hand the global domain tested with a texts that is trained, how we can see from Table 3.6 and Table 3.2 perform even better results that specific trained models.

Next experiment that we do is the same like the previous one, but now the domain is ”SPORT”. Training time for this model was 93.0 seconds in total, but 82.97 seconds spend on optimization. This model and test file, how in previous one is run with 300 abstracts. Table 3.12 shows the outcome of the experiment in coarse grained. From the table we can see that this domain provide a better result that ”POLITICS” domain, because here we have less annotated words. But, when compared with base experiment from Table 3.2 and Table 3.4 those experiments perform better results that this one.

Entity	P	R	F1
SPORT	0,9432	0,8839	0,9126
Totals	**0,9432**	**0,8839**	**0,9126**

Table 3.12: Outcomes of ”SPORT” base model in coarse grained run with ”SPORT” abstracts

Also we train a model in fine grained, with total time of 554.9 seconds, with 543.55 seconds spend on optimization and provide an experiment. Table 3.13 show that the result is little bit more better that result with model in coarse grained, but still this result is lower that the results for Table 3.6 and Table 3.8.

Entity	**P**	**R**	**F1**
Athlete	0,9713	0,8366	0,8989
Coach	1,0000	0,7500	0,8571
SportsClub	0,9453	0,9041	0,9242
SportsEvent	1,0000	0,7879	0,8814
SportsLeague	0,9418	0,8958	0,9182
SportsManager	1,0000	0,9615	0,9804
SportsTeam	0,9845	0,8301	0,9007
Totals	**0,9592**	**0,8750**	**0,9152**

Table 3.13: Outcomes of ”SPORT” base model in fine grained run with ”SPORT” abstracts

After provided 2 experiments with trained models for specific domain, the results shows that training a global model will perform better result than training a domain specific model.

Final experiment that we do with this size of abstracts (300 abstracts) is with ”TRANSPORTATION” domain. We needed 58.9 seconds to train the model, from which 50.35 seconds on optimization. Table 3.14 show the experiment outcome in coarse grained, where we can see that this result is lower than results from experiments provided in Table 3.2 and Table 3.5.

Entity	**P**	**R**	**F1**
TRANSPORTATION	0,9583	0,9109	0,9340
Totals	**0,9583**	**0,9109**	**0,9340**

Table 3.14: Outcomes of ”TRANSPORTATION” base model in coarse grained run with ”TRANSPORTATION” abstracts

Finally we make an experiment in fine grained. Total training time was 702.6 seconds, from which 686.50 seconds spend on optimization. In Table 3.15 we can see the results of provided experiment, where those results are even worse that the experiment with coarse grained model, which in previous two domain, ”SPORT” and ”POLITICS” was not that case. Also those results are worse than the experiments with a global domain in Table 3.6 and Table 3.9.

Entity	**P**	**R**	**F1**
Aircraft	0,9659	0,8333	0,8947
Automobile	1,0000	0,8000	0,8889
Infrastructure	0,9550	0,9550	0,9550
PublicTransitSystem	0,9662	0,9309	0,9482
Ship	1,0000	0,6429	0,7826
SpaceShuttle	1,0000	0,8333	0,9091
SpaceStation	0,0000	1,0000	0,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9660**	**0,9010**	**0,9324**

Table 3.15: Outcomes of ”TRANSPORTATION” base model in fine grained run with ”TRANSPORTATION” abstracts

In conclusion with the provided experiments in this section, we can say that training a global model and providing a NER is a better, but a little bit slowest solution than training a domain specific model, except the ”POLITICS” domain, where the results was better in domain specific model unlike the experiment with a global domain and test file with ”POLITICS” abstracts, but worse than experiment with a global domain tested with abstracts from all 3 domains. Then we waned to see the impact of fine grained trained models, where in most of the cases this kind of models provide a better results than models trained in coarse grained, except the experiment in ”TRANSPORTATION” specific domain where the coarse grained model was better that fine grained model.

After we finish with the main experiment, we were interested about the impact of the size of abstracts that will be used for training models. Next two subsections show the behavior of trained models.

3.3.2 Experiments that has less than 300 abstracts in model

In this subsection we want to see the behavior of models that are trained with less than 300 abstracts. First experiment is trained with 10 abstracts, then we have experiments with 20 abstracts, next experiment with 40 abstracts, and finally experiment with 100 abstracts. The order of abstracts, how we said earlier, is based on PageRank.

Model trained with 10 abstracts to every domain. To retrieve links from DBpedia with SPARQL and prepare data to be able to train models with 10 abstract, our algorithm explain in Section 2.4 takes in total 10.81 minutes, which comparing with main experiment, where we need 2.40 hours, is way more faster to prepare data. Of coarse this indicates that training models will also be faster than in main experiment.

Coarse grained model. How in the main experiment, also here we start with model trained in coarse grained. To train this kind of model we need 19.6

seconds, from which 17.44 seconds spend on optimization. From Table 3.16 we see that trained model perform the best results without any loosing of words in ”SPORT” and ”TRANSPORTATION” domains, but worst result in ”POLITICS” domain. The result of ”POLITICS” domain is even worst than the result from main experiment provided in Table 3.2. Because of this, there is a little bit lower overall result than in the main experiment. This can indicates that training models with lowest number of abstracts, for this kind of domains, is not worth. But let’s see how model will behaves when is tested with abstracts from a specific domains.

Entity	**P**	**R**	**F1**
POLITICS	0,9655	0,9333	0,9492
SPORT	1,0000	1,0000	1,0000
TRANSPORTATION	1,0000	1,0000	1,0000
Totals	**0,9811**	**0,9630**	**0,9720**

Table 3.16: Outcomes of global model in coarse grained run with 10 abstracts from every domain

Table 3.17 show the output of experiment where we have global model that is tested with 10 abstracts from ”POLITICS” domain. Here model do not annotated any word from other domains unlike in the main experiment in Table 3.3, but even this and the fact that here are much less abstracts does not help to provide a better results.

Entity	**P**	**R**	**F1**
POLITICS	0,9655	0,3636	0,5283
Totals	**0,9655**	**0,3636**	**0,5283**

Table 3.17: Outcomes of global model in coarse grained run with 10 abstracts from ”POLITICS” domain

Then we test our global model with abstracts from ”SPORT” domain. Table 3.18 show the outcome of the experiment. We can see that model perform perfect result, how in experiment in Table 3.16 without any misleading annotations, which we cannot say for the main experiment where model annotated words from ”POLITICS” and ”TRANSPORTATION” domains.

Entity	**P**	**R**	**F1**
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.18: Outcomes of global model in coarse grained run with 10 abstracts from ”SPORT” domain

Finally we tested the model with abstract from ”TRANSPORTATION” domain. From Table 3.19 we can see that model as well as in previous experiment perform maximum result without misleading annotations, unlike the main experiment where how we can see from Table 3.5 model annotate words with ”SPORT” domain and has a lowest result than this one.

Entity	P	R	F1
TRANSPORTATION	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.19: Outcomes of global model in coarse grained run with 10 abstracts from ”TRANSPORTATION” domain

In conclusion with the results from provided experiments we see that there is a huge impact of the number of abstracts for training a global model in coarse grained. We see that for ”SPORT” and ”TRANSPORTATION” domain our model provide maximum results, which is what we want to reach.

Fine grained model. After we finish the experiments with global models in coarse grained, we waned to see the impact of fine grained model. Does also here this kind of model will perform better results as was the case in main experiment, where global fine grained model perform a slide better results.

Training a fine grained model takes in total 124.7 seconds, from which 120,73 seconds spent in optimization. From Table 3.20 we can see that now fine grained model provide exactly the same overall result as well as coarse grained model. Also from table we can see in which ontology type our model fails to perform maximum result. So, because of PoliticalParty type where we have a lowest result, the total result is not at the maximum level, even thought other ontology types has maximum annotation.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
PoliticalParty	0,9600	0,9231	0,9412
Politician	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**0,9811**	**0,9630**	**0,9720**

Table 3.20: Outcomes of global model in fine grained run with 10 abstracts from every domain

Then how in previous experiments, we take the global train model and test it with abstracts from every specific domain separately. The first domain abstracts was from ”POLITICS” domain, where from Table 3.21 we can see that model perform same result as well as in coarse grained model experiment in Table 3.17. Also even our model and test files has words with Election ontology type, the model do not recognize any of them. With that misleading we have lower results, if that doesn't happens the model will perform pretty mach good recognition.

Entity	**P**	**R**	**F1**
Election	0,0000	0,0000	0,0000
PoliticalParty	0,9600	0,9231	0,9412
Politician	1,0000	1,0000	1,0000
Totals	**0,9655**	**0,3636**	**0,5283**

Table 3.21: Outcomes of global model in fine grained run with 10 abstracts from ”POLITICS” domain

Then we test it our model with abstracts from ”SPORT” domain. Table 3.22 shows that our model recognize all annotated words from test file without any misleading and perform maximum F1 score. In comparing with the main experiment in Table 3.8 where we have some loosing, here that is not the case and it is what we want to reach.

Entity	P	R	F1
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.22: Outcomes of global model in fine grained run with 10 abstracts from ”SPORT” domain

Final experiment that we do with global train model was with ”TRANSPORTATION” domain abstracts. In Table 3.23 we can see that model, same as in previous experiment with ”SPORT” abstracts, perform maximum F1 score result, which in comparing with the main experiment from Table 3.9 here we have improvements on result.

Entity	P	R	F1
Aircraft	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.23: Outcomes of global model in fine grained run with 10 abstracts from ”TRANSPORTATION” domain

In conclusion from the provided experiments where we had 10 abstracts on every domain, in comparing with the main experiments, we can say that there is an impact on performing a NER with a smallest number of abstracts for training a testing models. Here when we use coarse of fine grained global model and test it with texts from specific domain, except the ”POLITICS” domain abstracts, on other two domain, model perform NER without any misleading, which is what we waned to reach. Also training such small models takes way more less time, than training a big models.

3.3.2.1 Evaluation of domain specific models

In next 6 experiments trained models has 10 domain specific abstracts per model and also test files have the same specification.

”POLITICS” specific domain. First domain that we provide an experiment was ”POLITICS” specific domain. To train this model we need 3.8 seconds, from which 2.56 seconds spent in optimization. Table 3.24 show the outcome of the experiment, where the result here is way better, than with comparing with main experiment in Table 3.10 and the experiment with global train model tested with ”POLITICS” domain specific text in Table 3.17.

Entity	**P**	**R**	**F1**
POLITICS	0,9737	0,9610	0,9673
Totals	**0,9737**	**0,9610**	**0,9673**

Table 3.24: Outcome of ”POLITICS” domain specific model in coarse grained run with 10 abstracts from the same domain

Because we want to know the impact when model is trained in fine grained, we make an experiment with fine grained model. For training this model we need 8.3 seconds, from which 6.99 seconds spent in optimization. From Table 3.25 we can see that this kind of model provide a higher result than coarse grained model from previous experiment. Also this result is better than result from main experiment in Table 3.11 and the result from the experiment where we tested the global trained model with domain specific text in Table 3.21.

Entity	**P**	**R**	**F1**
Election	1,0000	0,9333	0,9655
PoliticalParty	0,9600	0,9231	0,9412
Politician	1,0000	1,0000	1,0000
Totals	**0,9867**	**0,9610**	**0,9737**

Table 3.25: Outcome of ”POLITICS” domain specific model in fine grained run with 10 abstracts from the same domain

From the ”POLITICS” domain specific experiments we can say that for this kind of domain with a lower number of abstracts for training a model the application provide NER with better results unlike the same experiments from the main experiment, where we have a worst results than here.

”SPORT” specific model. Training for a ”SPORT” domain coarse grained model with 10 abstracts we need 5.0 seconds, from which 3.50 seconds spend on optimization. From Table 3.26 is clear that this model, same as experiment in Table 3.18, provide excellent result unlike the main experiment in Table 3.12 where we have some loosing in recognition.

Entity	**P**	**R**	**F1**
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.26: Outcome of ”SPORT” domain specific model in coarse grained run with 10 abstracts from the same domain

Same as in the previous experiment, also here we have a fine grained model. Time needed for training this model was 26.4 seconds, from which 24.64 seconds spent in optimization. From Table 3.27 we see that the result is exactly the same like in coarse grained model (see Table 3.26) and the experiment from Table 3.22 where model is trained with abstracts from every domain and test file contains only abstracts from ”SPORT” domain. Those results from Table 3.27 are of coarse better that the results from the main experiment in Table 3.13, because here we don’t have any false recognition.

Entity	**P**	**R**	**F1**
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.27: Outcome of ”SPORT” domain specific model in fine grained run with 10 abstracts from the same domain

From the experiments provided in Table 3.26 and Table 3.27 as well as in previous experiment the number of abstracts needed for training a model plays significant role also in ”SPORT” domain. Here as well there is no difference if model is trained in coarse or fine grain, because we have the same result, but here plays role the time needed for training those models.

”TRANSPORTATION” specific model. Finally we have ”TRANSPORTATION” domain. For training a coarse grain model we needed 4.3 seconds, from which 3.10 seconds spent in optimization. From Table 3.28 we can see that as well as in ”SPORT” specific model NER is provided without any wrong recognition, which was also the case in experiment from Table 3.19. This means that this model also turned out to be better than the main experiment who has 300 abstracts (see Table 3.14.

Entity	**P**	**R**	**F1**
TRANSPORTATION	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.28: Outcome of ”TRANSPORTATION” domain specific model in coarse grained run with 10 abstracts from the same domain

With total time of 14.3 seconds, from which 12.99 seconds spent on optimization we trained a fine grained model. Table 3.29 shows that this model 100% precise same as the coarse grain model (see Table 3.29) and the experiment where model is trained with abstracts from every domain and test file contains only abstract from ”TRANSPORTATION” domain (see Table 3.23). Of coarse this experiment provide a better result that the main experiment in Table 3.15.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.29: Outcome of ”TRANSPORTATION” domain specific model in fine grained run with 10 abstracts from the same domain

”TRANSPORTATION” domain model as well as previous two domain models provide a better NER than the main experiment, but of coarse here we have much less annotated words in model and if we test this model with some other data, the results will be worst than in the main experiment where we have much more data.

3.3.2.2 Evaluation of global domain with 20 abstracts from every domain

Datasets with 20 abstracts for every domain. Because we waned to know the impact of train data, we decided to increase retrieved abstract from DBpedia to 20 abstracts per domain. Time need to retrieved those abstracts and prepare datasets for training in Stanford NER was 21.30 minutes.

Description of the experiment: In Table 3.30 we provide an experiment where the model was trained with abstracts from every domain, in total 60 abstracts, annotated in coarse grain. We need 36.6 seconds to train the model, from which 32.52 seconds spent in optimization. The model was tested with the same dataset that was trained.

Results of the experiment: Table 3.30 show the output of the experiment, where the overall precision is on maximum level, the recall on ”POLITICS” entities is a little bit lower, which results with overall lower recall and not bad at all F1 overall score. For the ”SPORT” and ”TRANSPORTATION” entities we have a maximum recognition. Referring to the main experiment from Table 3.2 is clearly that results here are better than in main experiment.

Entity	P	R	F1
POLITICS	1,0000	0,9615	0,9804
SPORT	1,0000	1,0000	1,0000
TRANSPORTATION	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9780**	**0,9889**

Table 3.30: Outcomes of global model in coarse grained run with 20 abstracts from every domain

Description of the experiment: For purposes of the experiment in Table 3.31 we have use the same global trained model from the previous experiment, but now the test file contains only 20 abstracts from ”POLITICS” domain.

Results of the experiment: From Table 3.31 we see that the result is way more worst than the previous experiment, but thanks to maximum precision and not recognizing any entity from other domains is slightly better than main experiment from Table 3.3 where model recognize entity from ”TRANSPORTATION” domain.

Entity	P	R	F1
POLITICS	1,0000	0,3906	0,5618
Totals	**1,0000**	**0,3906**	**0,5618**

Table 3.31: Outcomes of global model in coarse grained run with 20 abstracts from ”POLITICS” domain

Description of the experiment: This experiment is almost identical like previous one, with only difference is test file, where now we tested with abstracts from ”SPORT” domain.

Results of the experiment: From Table 3.32 we see that model provide maximum recognition without any wrong entity recognition of other domains. But this is not the case in the main experiment from Table 3.4 where also recognize entities from other two domains, even the file contains only abstracts from ”SPORT” domain.

Entity	P	R	F1
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.32: Outcomes of global model in coarse grained run with 20 abstracts from ”SPORT” domain

Description of the experiment: The final experiment is also the same like previous two, where now test file contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: Table 3.33 shows that for ”TRANSPORTATION” entities we have maximum recognition, but model also make a wrong entity recognition from ”SPORT” domain. This makes overall result not to be on his maximum and also in comparing with the main experiment from Table 3.5 where the model also recognize entity from ”SPORT” domain, here the overall result is worst.

Entity	**P**	**R**	**F1**
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	1,0000	1,0000	1,0000
Totals	**0,9231**	**1,0000**	**0,9600**

Table 3.33: Outcomes of global model in coarse grained run with 20 abstracts from ”TRANSPORTATION” domain

Description of the experiment: Experiment in Table 3.34 is provided with same data like the experiment in Table 3.30, but now the model and test data are annotated in fine grained. We needed in total 239.1 seconds to train model, from which 233.18 seconds spent in optimization.

Results of the experiment: How we can see from Table 3.34 our model provide maximum precision, but because there are 2 ontology types from ”TRANSPORTATION” domain, where out model provide a half on the maximum in the recall we have a lower result at the end. Also in comparing with the main experiment from Table 3.6 we have a slightly lower results here. As well those results are lower than the experiment in coarse grain (see Table 3.30).

Entity	P	R	F1
Aircraft	1,0000	0,5000	0,6667
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,5000	0,6667
PoliticalParty	1,0000	0,9512	0,9750
Politician	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9560**	**0,9775**

Table 3.34: Outcomes of global model in fine grained run with 20 abstracts from every domain

Description of the experiment: In this experiment we use the same trained model from previous experiment, but now the test file contains only abstracts from ”POLITICS” domain, who is annotated in fine grain.

Results of the experiment: Table 3.35 show the output of the experiment, where we can see that even we have Election type on model and test file, the model do not find any entity with that type. Also for the Politician type we have a very low recall, which reflects that there is a very low overall result. In comparing with the experiment in coarse grain (see Table 3.31 we have exactly the same overall result, but when we compare with the main experiment from Table 3.7 even in that experiment model also annotate some words from other two domains, the overall result is better than the result here.

Entity	P	R	F1
Election	0,0000	0,0000	0,0000
PoliticalParty	1,0000	0,9512	0,9750
Politician	1,0000	0,2000	0,3333
Totals	**1,0000**	**0,3906**	**0,5618**

Table 3.35: Outcomes of global model in fine grained run with 20 abstracts from ”POLITICS” domain

Description of the experiment: How in the previous experiment also here we have the same model but now tested with abstracts from ”SPORT” domain annotated in fine grain.

Results of the experiment: In Table 3.36 we have the output of the provided experiment. How we can see the results are excellent, there is no any wrong recognition or some lower values on precision and recall. Which in comparing with experiment in coarse grain (see Table 3.32) we have the same overall result, but we cannot say that about the results from the main experiment provided in Table 3.8 where we have wrong recognition of entities from other 2 domains and only one type has maximum precision and recall.

Entity	**P**	**R**	**F1**
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.36: Outcomes of global model in fine grained run with 20 abstracts from "SPORT" domain

Description of the experiment: The last experiment with the model used in previous 3 experiment is now the dataset test file with abstracts from "TRANSPORTATION" domain also annotated in fine grain.

Results of the experiment: Table 3.37 shows that our train model provide the same results how in experiment in Table 3.34 for the "TRANSPORTATION" ontology types. Also we have exactly the same overall result with the experiment in coarse grain (see Table 3.33, but when we compare results with the main experiment from Table 3.9 we have a way more better results than here, even thought that there model recognize some entities from other two domains.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	0,5000	0,6667
Infrastructure	1,0000	0,5000	0,6667
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsTeam	0,0000	1,0000	0,0000
Totals	**0,9091**	**0,8333**	**0,8696**

Table 3.37: Outcomes of global model in fine grained run with 20 abstracts from "TRANSPORTATION" domain

3.3.2.3 Evaluation of domain specific models with 20 abstracts

Description of the experiment: Experiment in Table 3.38 was provided with model trained only with abstracts from ”POLITICS” domain in coarse grain. To train this model we needed 5.4 seconds, from which 3.81 seconds spent in optimization.

Results of the experiment: How we can see from Table 3.38 the result is not bad at all. In comparing with experiment from Table 3.31 the results now are way more better and are more usable. Also referencing to main experiment from Table 3.10 where the only difference is the number of abstracts used for training the model, now the result is a little bit better than there.

Entity	**P**	**R**	**F1**
POLITICS	0,9921	0,9766	0,9843
Totals	**0,9921**	**0,9766**	**0,9843**

Table 3.38: Outcome of ”POLITICS” domain specific model in coarse grained run with 20 abstracts from the same domain

Description of the experiment: For the purposes of this experiment we have used the same data from the previous one, but now annotated in fine grain. To train this kind of model we needed 12.4 seconds, from which 10.80 seconds spent in optimization.

Results of the experiment: We tested the model with the same data that was created and how we can see from Table 3.39 model provides maximum precision on entities, but because of lower recall we have overall a quite lower F1 score. But in comparing with previous experiment the result is slightly better, which we cannot say that about experiment in Table 3.35 where result is terrible. Also in comparing with the main experiment from Table 3.11 now model provides also a little bit better result, but not that significant like in experiment from Table 3.35

Entity	**P**	**R**	**F1**
Election	1,0000	0,9688	0,9841
PoliticalParty	1,0000	0,9512	0,9750
Politician	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9766**	**0,9881**

Table 3.39: Outcome of ”POLITICS” domain specific model in fine grained run with 20 abstracts from the same domain

Description of the experiment: Experiment in Table 3.40 is provided with a coarse grain model trained with abstracts only from ”SPORT” domain. The time needed to train this model was 5.0 seconds, from which 3.50 seconds spent in optimization. To test it we have used the same dataset that model was trained.

Results of the experiment: How we can see from Table 3.40 the trained model provide excellent recognition on entities from test dataset, without any miss or wrong recognition. The same results we have in experiment with a global domain model tested with the same dataset (see Table 3.32). But when we compare the results from here and the results from the main experiment (see Table 3.12) we see that now results are better, but here we have a less entities.

Entity	**P**	**R**	**F1**
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.40: Outcome of ”SPORT” domain specific model in coarse grained run with 20 abstracts from the same domain

Description of the experiment: In this experiment we have used the same dataset from previous, but now entities are annotated in fine grain. To train a fine grain model we have need 26.4 seconds, from which 24.46 second spent in optimization. As well as previous the model is tested with the dataset that is trained.

Results of the experiment: In Table 3.41 we see the output of the experiment. How in the previous experiment in coarse grain, also here the results are excellent without any looseness of unrecognized entities. As well we have the same result in Table 3.36 where we have a global domain and the same test file (test file contains only abstracts from ”SPORT” domain). But in comparing with the main experiment from Table 3.13, again the results there are lower than here (experiment from Table 3.41).

Entity	**P**	**R**	**F1**
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.41: Outcome of ”SPORT” domain specific model in fine grained run with 20 abstracts from the same domain

Description of the experiment: At the end we train a model with abstracts from ”TRANSPORTATION” domain. To train a coarse grain model from this domain we needed 4.3 seconds, from which 3.10 seconds spent in optimization. Of coarse, here also we tested the model with the same dataset that was created.

Results of the experiment: In Table 3.42 we see that the model provide maximum precision, but lower recall on entities which results with a worst F1 score. Surprisingly result here is lower than the experiment where we had a global model tested with the same dataset like here (see Table 3.33), even those that there we have a wrong entity recognition from "SPORT" domain, the results is still better. As well the results from the main experiment in Table 3.14 are better than here, which was not the case in the previous 4 experiments.

Entity	**P**	**R**	**F1**
TRANSPORTATION	1,0000	0,8333	0,9091
Totals	**1,0000**	**0,8333**	**0,9091**

Table 3.42: Outcome of "TRANSPORTATION" domain specific model in coarse grained run with 20 abstracts from the same domain

Description of the experiment: We also train a fine grain model from "TRANSPORTATION" domain. To train it we needed 14.3 seconds, from which 12.99 seconds spent in optimization. As well as the previous experiment, the model is tested with the same dataset that is created.

Results of the experiment: The output of the experiment seen in Table 3.43 are quite surprisingly. Until now in the experiments with a lower abstracts than the main experiment, the fine grained models provides same or better results than coarse grained model. Here model provides a worst result than the previous experiment. Also experiment in Table 3.37 and the main experiment from Table 3.15 have better results than here.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	0,5000	0,6667
Infrastructure	1,0000	0,5000	0,6667
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,7500**	**0,8571**

Table 3.43: Outcome of "TRANSPORTATION" domain specific model in fine grained run with 20 abstracts from the same domain

Until now we can say that the number of abstracts used to train a models has an impact on the final results and also it is faster to train a smaller models, but in that case we are short on entities and types. We still have 2 more groups of different number of abstracts, 40 abstracts per domain and 100 abstracts per domain. So let see how those groups will behave in comparing with the main experiment, where we have 300 abstracts per domain.

3.3.2.4 Evaluation of global domain with 40 abstracts from every domain

Datasets with 40 abstracts for every domain. Now we have increased number of retrieved links from DBpedia to 40 abstracts. To retrieve links and prepare datasets that contains 40 abstracts to every domain our algorithm needs 30.94 minutes.

Description of the experiment: In Table 3.44 we provide an experiment where the model is trained with all retrieved abstracts (120 abstracts in total) in coarse grain. To train this model with Stanford NER we needed 101.7 seconds, from which 91.52 seconds spent in optimization. We have tested the model with the same dataset that was created.

Results of the experiment: Table 3.44 shows the output of the experiment, where we see that for the "SPORT" domain we have maximum results, but also results from other domains are not bad at all. This gives a very good total results on precision, recall and F1 score. In comparing with the main experiment from Table 3.2 now we have a little bit more better results, but with a lower number of entities in model.

Entity	**P**	**R**	**F1**
POLITICS	0,9890	0,9375	0,9626
SPORT	1,0000	1,0000	1,0000
TRANSPORTATION	1,0000	0,9846	0,9922
Totals	**0,9960**	**0,9724**	**0,9841**

Table 3.44: Outcomes of global model in coarse grained run with 40 abstracts from every domain

Description of the experiment: In this experiment we have used the same model from previous, but now it is tested with dataset that contains only abstracts from "POLITICS" domain.

Results of the experiment: How we see from Table 3.45 even those that we don't have any recognition from other domains and a maximum precision, the recall is very low which reflects with low F1 score. When we compare with the previous experiment we see that there result for "POLITICS" domain is better than here. Also in comparing with the main experiment from Table 3.3 where model recognize some entities from "TRANSPORTATION" domain and precision is lower, still those results are better than here.

Entity	P	R	F1
POLITICS	0,9890	0,3529	0,5202
Totals	**0,9890**	**0,3529**	**0,5202**

Table 3.45: Outcomes of global model in coarse grained run with 40 abstracts from ”POLITICS” domain

Description of the experiment: For purposes of this experiment we also used the global modal, but now it is tested with dataset that contains only abstracts from ”SPORT” domain.

Results of the experiment: How we can see from Table 3.46 model provides perfect recognition without any wrong entity recognition from other domains. The same result for ”SPORT” domain we have in experiment from Table 3.44, but in comparing with the main experiment from Table 3.4 where model recognize some entities from other two domain, as well the result for ”SPORT” domain are lower than here.

Entity	P	R	F1
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.46: Outcomes of global model in coarse grained run with 40 abstracts from ”SPORT” domain

Description of the experiment: Finally we tested the global domain with abstracts from ”TRANSPORTATION” domain.

Results of the experiment: As we can see from Table 3.47 model provides maximum precision on ”TRANSPORTATION” domain, but a little bit lower recall, which of coarse reflects on F1 score. Model also recognize some wrong entities from ”SPORT” domain which results with lower total F1 score. As well here we have same results like in experiment from Table 3.44. In comparing with the main experiment from Table 3.5 where model also recognizes wrong entities from ”SPORT” domain the overall result is slightly better than here.

Entity	P	R	F1
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	1,0000	0,9846	0,9922
Totals	**0,9697**	**0,9846**	**0,9771**

Table 3.47: Outcomes of global model in coarse grained run with 40 abstracts from ”TRANSPORTATION” domain

Description of the experiment: For the purposes of this experiment we train a fine grain model with abstracts from every domain. The time that we needed to train this model was 287.5 second, from which 278.99 seconds spent in optimization. The model is tested with the same dataset that is created.

Results of the experiment: How we can see from Table 3.48 the list of ontology types now is longer than in previous two groups of experiment (with 10 and 20 abstracts). Also model provides maximum precision on every type except PoliticalParty type and a lower recall on the same type as well the Politician ontology type. This results with a lower overall result on every measurement. But in comparing with the model in coarse grain (see Table 3.44), here the overall result is better, but not what significant. In comparing with the main experiment from Table 3.6 where model has more ontology types, the overall result is lower than now, but the difference in the results is not that big. Another thing is that even test dataset contains Election type, model do not find any entity of that type.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
Infrastructure	1,0000	1,0000	1,0000
PoliticalParty	0,9863	0,9730	0,9796
Politician	1,0000	0,8182	0,9000
PublicTransitSystem	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**0,9960**	**0,9764**	**0,9861**

Table 3.48: Outcomes of global model in fine grained run with 40 abstracts from every domain

Description of the experiment: In this experiment we have used the same model from previous, but now it is tested with the dataset that contains only abstracts from "POLITICS" domain, of coarse annotated in fine grain.

Results of the experiment: From Table 3.49 we see that model almost fail the test, even thought that test data are part of training the model. In test dataset we have entities with Election ontology type, but model do not recognize any of them, as well on Politician type we have maximum precision and a very low recall which reflects with low F1 score. Those results are the same with the experiment in coarse grain from Table 3.45. As we compare the results for every ontology type from previous experiment we will get way more better results, also the results from the main experiment in Table 3.7 are a slightly better although model recognize types of entities from other domains.

Entity	**P**	**R**	**F1**
Election	0,0000	0,0000	0,0000
PoliticalParty	0,9863	0,9730	0,9796
Politician	1,0000	0,1565	0,2707
Totals	**0,9890**	**0,3529**	**0,5202**

Table 3.49: Outcomes of global model in coarse grained run with 40 abstracts from ”POLITICS” domain

Description of the experiment: Experiment here has the same trained model like the previous one, with the difference that now it is tested with dataset that contains abstracts only from ”SPORT” domain.

Results of the experiment: The output of the experiment shown in Table 3.50 is exactly that we waned to reach. Model provide maximum results on every ontology type without any wrong recognition. The same results for ”SPORT” ontology types we have in experiment with the global dataset in Table 3.48 and the experiment in coarse grain from Table 3.46. We cannot say that about the main experiment in Table 3.8 where some ontology types don’t have maximum results and also model recognize entities from other two domains, which results with lower total result than here.

Entity	**P**	**R**	**F1**
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	1,0000	1,0000
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.50: Outcomes of global model in coarse grained run with 40 abstracts from ”SPORT” domain

Description of the experiment: The final experiment with the fine grain global model that we provide was that now it is tested with the abstracts from ”TRANSPORTATION” domain.

Results of the experiment: As we can see from Table 3.51, model provide maximum results for ontology types from ”TRANSPORTATION” domain, but also recognize some entities from ”SPORT” domain, which was not in test dataset. Because of that the total result is a little bit lower that the maximum. When we compare the results for every type with the results from experiment in Table 3.48 we can see that are the same. Also this experiment provides better result that the coarse grain experiment from Table 3.47. A little bit surprisingly is that, that the main experiment from Table 3.9 gives better results than the experiment here.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Infrastructure	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	1,0000	1,0000
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsTeam	0,0000	1,0000	0,0000
Totals	**0,9701**	**1,0000**	**0,9848**

Table 3.51: Outcomes of global model in coarse grained run with 40 abstracts from ”TRANSPORTATION” domain

3.3.2.5 Evaluation of domain specific models with 40 abstracts

Description of the experiment: The experiment here is provided with coarse grain model that is trained with dataset that contains only abstracts from ”POLITICS” domain. To train this model we needed 13.9 seconds, from which 10.36 seconds spent in optimization. It is tested with the same dataset that is trained.

Results of the experiment: Table 3.52 shows the outcome of the experiment, where in comparing with experiment from Table 3.44 we can see a big improvement on the result, where now we are closer to the best performance. In comparing with the main experiment from Table 3.10, now we have a way more better result than there.

Entity	**P**	**R**	**F1**
POLITICS	0,9921	0,9804	0,9862
Totals	**0,9921**	**0,9804**	**0,9862**

Table 3.52: Outcome of ”POLITICS” domain specific model in coarse grained run with 40 abstracts from the same domain

Description of the experiment: This experiment is provided with same data like the previous one, but now dataset is annotated in fine grain. To train a ”POLITICS” fine grain model we needed 41.0 seconds, from which 36.11 seconds spent in optimization. Of coarse it is tested with the dataset that is trained.

Results of the experiment: As we can see from Table 3.53 the result is better than the previous experiment, and also model recognize all entities from the domain, which was not the case in experiment from Table 3.49 and as well the result now is way more better. Comparing with the main experiment from Table 3.11, now we again have a better results to every ontology type, and of coarse better total result.

Entity	P	R	F1
Election	1,0000	0,9848	0,9924
PoliticalParty	0,9863	0,9730	0,9796
Politician	1,0000	1,0000	1,0000
Totals	**0,9960**	**0,9882**	**0,9921**

Table 3.53: Outcome of "POLITICS" domain specific model in fine grained run with 40 abstracts from the same domain

Description of the experiment: For the purposes of this experiment we have used dataset with abstracts from "SPORT" domain annotated in coarse grain. To train a coarse grain model we needed 10.0 seconds, from which 6.84 seconds spent in optimization. Here as well we test the model with the dataset that is trained.

Results of the experiment: Table 3.55 show the output of the experiment, where we have a maximum entity recognition, which was also the case in experiment with a global model and same test dataset like here (see Table 3.46. In comparing with the same experiment in Table 3.12 from main experiment, is clearly that now we had a way more better results than there.

Entity	P	R	F1
SPORT	1,0000	1,0000	1,0000
Totals	**1,0000**	**1,0000**	**1,0000**

Table 3.54: Outcome of "SPORT" domain specific model in coarse grained run with 40 abstracts from the same domain

Description of the experiment: Then we used same data like in previous experiment, but now dataset is annotated in fine grain. To train a "SPORT" fine grain domain model with Stanford NER we needed 56.2 seconds, from which 52.77 seconds spent in optimization. It is tested as always, with the same dataset that is trained.

Results of the experiment: In Table 3.55 we see the experiment outcome, where because of the lower recall on SportsClub ontology type, the overall result is slightly lower. So after some time, again fine grain model provides a little bit worst result than coarse grain model. Because of this the experiment with a global model and same test dataset like here from Table 3.50 gives better results. For consolation is the fact that model here provide significantly better result than the main experiment in Table 3.13

Entity	P	R	F1
Athlete	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	0,9474	0,9730
SportsEvent	1,0000	1,0000	1,0000
SportsLeague	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9890**	**0,9945**

Table 3.55: Outcome of "SPORT" domain specific model in fine grained run with 40 abstracts from the same domain

Description of the experiment: The final experiments with this number of abstracts is with "TRANSPORTATION" domain. To train a coarse grain model we needed 9.1 seconds, from which 6.24 seconds spent in optimization. As in all previous experiment, test is provided with the same dataset that is trained.

Results of the experiment: Table 3.57 shows the output of the coarse grain model experiment, where the result is very close to the maximum. In experiment where we use a global model who is tested with same dataset as here (see Table 3.47), overall results there is lower, because model recognize some entities from another domain, but the result from "TRANSPORTATION" domain is the same as here. As well this experiment provides a better result than the main experiment from Table 3.14.

Entity	P	R	F1
TRANSPORTATION	1,0000	0,9846	0,9922
Totals	**1,0000**	**0,9846**	**0,9922**

Table 3.56: Outcome of "TRANSPORTATION" domain specific model in coarse grained run with 40 abstracts from the same domain

Description of the experiment: Lastly we train a fine grain model with the same data like in previous experiment, but of coarse annotated in fine grain. To train this kind of model we needed 44.6 seconds, from which 41.63 seconds spent in optimization. It is tested as usual.

Results of the experiment: As we can see from Table 3.57 we have exactly the same result like in coarse grain experiment. In comparing with the experiment in Table 3.51 where because of wrong entity recognition model gives lower results than here, although the results on every ontology type is the same, except the PublicTransitSystem type, where now we had a lower recall. Also model here, again gives us better results than the main experiment in Table 3.15.

Entity	P	R	F1
Aircraft	1,0000	1,0000	1,0000
Infrastructure	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	0,9630	0,9811
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9846**	**0,9922**

Table 3.57: Outcome of ”TRANSPORTATION” domain specific model in fine grained run with 40 abstracts from the same domain

3.3.2.6 Evaluation of global domain with 100 abstracts from every domain

Datasets with 100 abstracts for every domain. In this final group of 100 abstracts per domain we will repeat all experiments like in previous groups, to see how model will behaves when is closer to the number of abstracts in main experiment. To retrieve and prepare datasets for training in Stanford NER, our algorithm needs in total 63.71 minutes, which is twice more than when we had 40 abstracts for every domain.

Description of the experiment: In Table 3.58 we provide an experiment where the model is trained with, in total, 300 abstracts annotated in coarse grain. To train this kind of model we needed 258.1 seconds, from which 246.28 seconds spent in optimization. The model is tested with the dataset that was trained.

Results of the experiment: As we can see from Table 3.58 now model provides maximum result only in precision measurement on ”TRANSPORTATION” domain. From here we can see that results are closer to the main experiment from Table 3.2, but still results here are little bit better than there.

Entity	P	R	F1
POLITICS	0,9920	0,9612	0,9764
SPORT	0,9963	0,9926	0,9944
TRANSPORTATION	1,0000	0,9735	0,9865
Totals	**0,9952**	**0,9766**	**0,9856**

Table 3.58: Outcomes of global model in coarse grained run with 100 abstracts from every domain

Description of the experiment: For purposes of this experiment we have used the model train previously, but now it is tested with the dataset that contains only abstracts from ”POLITICS” domain, annotated in coarse grain.

Results of the experiment: From Table 3.59 we see that even we have a precision close to maximum value, but because of very low recall, the F1 score is lower, which means that model recognize only half of our entities. When we compare with the previous experiment is clearly that there the results for ”POLITICS” domain are better. Also comparably with the main experiment from Table 3.3 results there a quite better than now, although there model recognize some entities from ”TRANSPORTATION” domain.

Entity	**P**	**R**	**F1**
POLITICS	0,9920	0,3615	0,5299
Totals	**0,9920**	**0,3615**	**0,5299**

Table 3.59: Outcomes of global model in coarse grained run with 100 abstracts from ”POLITICS” domain

Description of the experiment: In this experiment we also used the global trained model, but now it is tested with ”SPORT” domain abstracts dataset.

Results of the experiment: Table 3.60 shows the output of the experiment, from where we see that even the results aren't at their maximum are quite satisfying, but are a little bit lower than in global experiment in Table 3.58. As well in comparing with the main experiment from Table 3.4, now, very importantly, we don't have any wrong entity recognition and even if we compare only results from ”SPORT” domain, still result is better now.

Entity	**P**	**R**	**F1**
SPORT	0,9962	0,9888	0,9925
Totals	**0,9962**	**0,9888**	**0,9925**

Table 3.60: Outcomes of global model in coarse grained run with 100 abstracts from ”SPORT” domain

Description of the experiment: Final experiment with the global domain is that is tested with the dataset that contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: From Table 3.61 we see that model also recognize some entities from ”SPORT” domain, that are not part of the tested dataset. Also if we compare only results for ”TRANSPORTATION” domain with the results from global experiment in Table 3.58, they are the same, but now because of that wrong recognition, overall results is slightly lower. The exact situation is in main experiment from Table 3.5 where model also make a mistake, but there results are better than here.

Entity	P	R	F1
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	1,0000	0,9735	0,9865
Totals	**0,9821**	**0,9735**	**0,9778**

Table 3.61: Outcomes of global model in coarse grained run with 100 abstracts from "TRANSPORTATION" domain

Description of the experiment: For purposes of this and the next 3 experiments we train a fine grain global model with 100 abstracts from every domain. To train this kind of model we needed 857.3 seconds, from which 844.91 seconds spent in optimization. In this experiment we tested the model with the dataset that was trained.

Results of the experiment: From Table 3.62 we can see the results for every particular ontology type. Now model preforms bad results mostly for types from "SPORT" domain. And it happens again that the fine grain model gives worst total results than the coarse grain model (see Table 3.58 for coarse grain results). As well referencing to main experiment in Table 3.6 the results there are way more better than here, although there we have more data.

Entity	P	R	F1
Aircraft	1,0000	0,6957	0,8205
Athlete	1,0000	0,4167	0,5882
Automobile	1,0000	1,0000	1,0000
Coach	1,0000	0,6667	0,8000
Infrastructure	1,0000	1,0000	1,0000
PoliticalParty	0,8774	0,6700	0,7598
Politician	1,0000	0,7455	0,8542
PublicTransitSystem	0,9744	0,7308	0,8352
Ship	1,0000	0,6000	0,7500
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,9512	0,9398	0,9455
SportsEvent	0,9737	0,8605	0,9136
SportsLeague	0,9500	0,8636	0,9048
SportsManager	1,0000	1,0000	1,0000
SportsTeam	1,0000	0,6364	0,7778
Train	1,0000	1,0000	1,0000
Totals	**0,9452**	**0,7535**	**0,8385**

Table 3.62: Outcomes of global model in fine grained run with 100 abstracts from every domain

Description of the experiment: In this experiment we tested the global model with dataset that contains only abstracts from ”POLITICS” domain, of coarse annotated in fine grain.

Results of the experiment: Table 3.63 shows the output of the experiment, where model provides terrible results and also recognize entity with SportEvent type which is not part of the dataset, but don’t recognize any entity with Election type who is part of the dataset. As well this experiment gives a worst total result than the coarse grain experiment in Table 3.59 and worst type results than global model from previous experiment. Finally the results now are worse than in main experiment from Table 3.7.

Entity	**P**	**R**	**F1**
Election	0,0000	0,0000	0,0000
PoliticalParty	0,8774	0,6700	0,7598
Politician	1,0000	0,1285	0,2278
SportsEvent	0,0000	1,0000	0,0000
Totals	**0,8985**	**0,2580**	**0,4009**

Table 3.63: Outcomes of global model in fine grained run with 100 abstracts from ”POLITICS” domain

Description of the experiment: Here we also have the global train model, but for this experiment is tested with dataset abstracts from ”SPORT” domain.

Results of the experiment: As we see from Table 3.64 model gives exact the same results for every ontology type like in global experiment in Table 3.62, where because of worst recall the overall result are lower than in comparing with the coarse grain experiment in Table 3.60. As well referencing to main experiment in Table 3.8, now again results are worse than there.

Entity	**P**	**R**	**F1**
Athlete	1,0000	0,4167	0,5882
Coach	1,0000	0,6667	0,8000
SportsClub	0,9506	0,9390	0,9448
SportsEvent	1,0000	0,8605	0,9250
SportsLeague	0,9500	0,8636	0,9048
SportsManager	1,0000	1,0000	1,0000
SportsTeam	1,0000	0,6250	0,7692
Totals	**0,9683**	**0,7985**	**0,8753**

Table 3.64: Outcomes of global model in fine grained run with 100 abstracts from ”SPORT” domain

Description of the experiment: The final experiment with the global model is with dataset that contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: Table 3.65 shows the outcome of the experiment, where again the results for every individual ontology type aren't increased from the result in global experiment in Table 3.62. Because of worst recall in Aircraft, PublicTransitSystem and Ship types we have a lower total result than in coarse grain experiment in Table 3.61. Not surprisingly for this group of experiment, the results from here are also worst than in main experiment from Table 3.9.

Entity	P	R	F1
Aircraft	1,0000	0,6957	0,8205
Automobile	1,0000	1,0000	1,0000
Infrastructure	1,0000	1,0000	1,0000
PublicTransitSystem	0,9744	0,7308	0,8352
Ship	1,0000	0,6000	0,7500
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,0000	1,0000	0,0000
SportsTeam	0,0000	1,0000	0,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9677**	**0,7965**	**0,8738**

Table 3.65: Outcomes of global model in fine grained run with 100 abstracts from "TRANSPORTATION" domain

3.3.2.7 Evaluation of domain specific models with 100 abstracts

Description of the experiment: In this experiment we train a domain specific coarse grain model with dataset that contains only abstracts from "POLITICS" domain. The time needed to train this model was 34.2 seconds, from which 28.45 seconds spent in optimization. The model is tested with the same dataset that is trained.

Results of the experiment: As we can see from Table 3.66 this model provides a better result than the 2 experiments with the global model in Table 3.58 and and a way more better results than experiment from Table 3.59. As well referencing to main experiment in Table 3.10, now model gives better results.

Entity	P	R	F1
POLITICS	0,9956	0,9898	0,9927
Totals	**0,9956**	**0,9898**	**0,9927**

Table 3.66: Outcome of "POLITICS" domain specific model in coarse grained run with 100 abstracts from the same domain

Description of the experiment: With the same data from the previous experiment, but not annotated in fine grain, we train new model. We needed 89.4 seconds to train this model, from which 83.54 seconds spent in optimization. The model is tested with the same dataset that was trained.

Results of the experiment: Table 3.67 shows the outcome of the experiment, where we see that even results on every type are close to maximum, the overall results is slightly lower than the previous experiment. But in comparing to the experiments with global domain in Table 3.62 and Table 3.63, now results are better, we don't have any wrong recognition and also entities from Election ontology type are recognized. Referencing to the main experiment in Table 3.11, now as well results are way more better and more usable.

Entity	**P**	**R**	**F1**
Election	1,0000	0,9878	0,9939
PoliticalParty	0,9950	0,9852	0,9901
Politician	0,9937	0,9906	0,9922
Totals	**0,9956**	**0,9883**	**0,9920**

Table 3.67: Outcome of "POLITICS" domain specific model in fine grained run with 100 abstracts from the same domain

Description of the experiment: For the purposes of the experiment we train a coarse grain model with dataset that contains only abstracts from "SPORT" domain. Time needed to train this kind of model was 25.3 seconds, from which 20.47 seconds spent in optimization. The model is tested like in previous experiments, with the same dataset that is trained.

Results of the experiment: From Table 3.68 we see that model gives a slightly better results than in experiments with global domain in Table 3.58 and Table 3.60. Also referencing to the main experiment in Table 3.12, now model gives results that are closer to the maximum values, but we don't have that much data here.

Entity	**P**	**R**	**F1**
SPORT	0,9963	0,9963	0,9963
Totals	**0,9963**	**0,9963**	**0,9963**

Table 3.68: Outcome of "SPORT" domain specific model in coarse grained run with 100 abstracts from the same domain

Description of the experiment: This experiment is provided with a "SPORT" fine grain model. To train this model we needed 194.2 seconds, from which 187.77 seconds spent in optimization. Test is the same like previous experiment, where dataset now are annotated in fine grain.

Results of the experiment: As we see from Table 3.64 the overall results is lower than previous experiment, but way better than experiments provided with the global model in Table 3.62 and Table 3.64. Also referencing to main experiment in Table 3.13, now model was more precise than there, an because of that gives better results.

Entity	**P**	**R**	**F1**
Athlete	1,0000	0,9722	0,9859
Coach	1,0000	1,0000	1,0000
SportsClub	1,0000	0,9878	0,9939
SportsEvent	1,0000	0,9767	0,9882
SportsLeague	1,0000	1,0000	1,0000
SportsManager	1,0000	1,0000	1,0000
SportsTeam	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9888**	**0,9944**

Table 3.69: Outcome of ”SPORT” domain specific model in fine grained run with 100 abstracts from the same domain

Description of the experiment: Experiment in Table 3.70 is provided with a coarse grain model train with dataset from ”TRANSPORTATION” domain. To train this model we needed 20.5 seconds in total, from which 14.93 seconds spent in optimization. As previous model is tested with same dataset that is trained.

Results of the experiment: Table 3.70 shows that model provides maximum precision on entities, but slight lower recall which results with lower F1 score. Comparably with experiments with global model in Table 3.58 and Table 3.61, now model gives better score than there and it's faster to train and provide experiment. As well referencing to main experiment in Table 3.14 again those results are lower than domain specific model.

Entity	**P**	**R**	**F1**
TRANSPORTATION	1,0000	0,9912	0,9956
Totals	**1,0000**	**0,9912**	**0,9956**

Table 3.70: Outcome of ”TRANSPORTATION” domain specific model in coarse grained run with 100 abstracts from the same domain

Description of the experiment: The final experiment with this group of number of abstracts is fine grain model for ”TRANSPORTATION” domain. Time taken to train this kind of model was in total 134.6 seconds, from which 126.98 seconds, spent in optimization. Testing routine is the same here.

Results of the experiment: From Table 3.71 we again have a lower result than the previous experiment with coarse grain model. But referencing to main experiment in Table 3.15, results now are closer to maximum value. As well the same situation is with experiments provided with global model in

Table 3.62 and Table 3.65 where those results there are lower than domain specific model results.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Automobile	1,0000	1,0000	1,0000
Infrastructure	1,0000	1,0000	1,0000
PublicTransitSystem	1,0000	0,9808	0,9903
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	0,6667	0,8000
SpaceStation	1,0000	1,0000	1,0000
Train	1,0000	1,0000	1,0000
Totals	**1,0000**	**0,9735**	**0,9865**

Table 3.71: Outcome of ”TRANSPORTATION” domain specific model in fine grained run with 100 abstracts from the same domain

In conclusion we can say that models that were trained with lower number of abstracts than in main experiment, mostly provides a better results, of coarse the time needed to train those models and make experiments was faster, but here we don't have that much data unlike in main experiment, which if we try some other dataset who was not used to train the models, the results can we worst here than in models who has more trained data. Except the experiment where we had 100 abstracts to every domain, in most of other experiment, the fine grain model gives more precise recognition than the coarse grain model, which for us was quite surprise. And in the every group of experiment the domain specific models gives same or better results than global models and if we put there the time needed for training or testing domain specific models, then the results for now is that domain specific models are better for usage.

3.3.3 Experiments that have more than 300 abstracts in model and test files

In this subsection we have a two groups of number of abstracts retrieved from DBpedia, that are bigger than in the main experiment. One of them is where we have 400 abstracts per domain and the other one is with 500 abstracts per domain, which is the maximum that we succeeded to train.

3.3.3.1 Evaluation of global domain with 400 abstracts from every domain

Datasets with 400 abstracts for every domain. As well because we waned to know the impact of train data who has more abstracts than the main experiment, we've increased the number of retrieved data to 400 abstracts per domain. Our algorithm needs 184.92 minutes to retrieve data from DBpedia and prepare datasets ready to use in Stanford NER application.

Description of the experiment: In Table 3.72 we provide an experiment where the model was trained with abstracts from every domain, in total 1200 abstracts, annotated in coarse grain. We need 1041.3 seconds to train the model, from which 1008.25 seconds spent in optimization. The model was tested with the same dataset that was trained.

Results of the experiment: Table 3.72 show the output of the experiment, where the results to every domain are close to the maximum values on every measurement. We can say that for such big model the results are fantastic. Referring to the main experiment from Table 3.2, now this kind of model provide slightly lower results, but we have more trained data here.

Entity	**P**	**R**	**F1**
POLITICS	0,9804	0,9434	0,9615
SPORT	0,9832	0,9590	0,9709
TRANSPORTATION	0,9941	0,9754	0,9847
Totals	**0,9849**	**0,9584**	**0,9714**

Table 3.72: Outcomes of global model in coarse grained run with 400 abstracts from every domain

Description of the experiment: For purposes of the experiment in Table 3.73 we have use the same global trained model from the previous experiment, but now the test file contains only 400 abstracts from ”POLITICS” domain.

Results of the experiment: From Table 3.73 we see that the result is way more worst than the previous experiment, also model recognizes entities from other domain which are not part of dataset.If we compare only result of ”POLITICS” type with the previous experiment we can see that now results are worst then there. Also in comparing with the main experiment from Table 3.3, now model gives a very very little better results, although recognize entities from ”SPORT” domain, which is not the case in main experiment.

Entity	**P**	**R**	**F1**
POLITICS	0,9754	0,4082	0,5756
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9531**	**0,4082**	**0,5716**

Table 3.73: Outcomes of global model in coarse grained run with 400 abstracts from ”POLITICS” domain

Description of the experiment: This experiment is almost identical like previous one, with only difference is test file, where now we tested with abstracts from ”SPORT” domain.

Results of the experiment: From Table 3.74 we see that model despite ”SPORT” type, recognize entities from other 2 domain which are not part of dataset, This brings results a little bit lower than the results of ”SPORT” type. We have the same situation on the main experiment in Table 3.4, but now results there are better for a bit. As well if we get only results for ”SPORT” type and compare with the results from global model in Table 3.72 we can see that now results are again a bit better.

Entity	**P**	**R**	**F1**
POLITICS	0,0000	1,0000	0,0000
SPORT	0,9837	0,9588	0,9711
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9805**	**0,9588**	**0,9695**

Table 3.74: Outcomes of global model in coarse grained run with 400 abstracts from ”SPORT” domain

Description of the experiment: The final experiment is also the same like previous two, where now test file contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: Table 3.75 shows that for ”TRANSPORTATION” entities we have precision closer to maximum value. But how in previous 2 experiment, also here model recognize entities which are not part of test dataset. Now if we compare only the results of ”TRANSPORTATION” type with the results in global model from Table 3.72 we see that global model provides a slight better results. Referring to the main experiment from Table 3.5, the overall results of the experiments are the same, but now model recognize entities also with ”POLITICS” type and the result for ”TRANSPORTATION” type are as well a bit lower.

Entity	**P**	**R**	**F1**
POLITICS	0,0000	1,0000	0,0000
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	0,9939	0,9762	0,9806
Totals	**0,9861**	**0,9822**	**0,9841**

Table 3.75: Outcomes of global model in coarse grained run with 400 abstracts from ”TRANSPORTATION” domain

Description of the experiment: Experiment in Table 3.76 is provided with same data like the experiment in Table 3.72, but now the model and test data are annotated in fine grained. We needed in total 5139.7 seconds to train model, from which 5095.94 seconds spent in optimization.

Results of the experiment: How we can see from Table 3.76 our model provide maximum precision on most of entity types and maximum recall on some entity types. This helps to have a bit better results the experiment in coarse grain model from Table 3.72. Also in comparing with the main experiment from Table 3.6 we have a slightly lower results here, but a more annotated entities.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Athlete	1,0000	0,9899	0,9949
Automobile	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9896	0,9948
PoliticalParty	0,9766	0,9486	0,9624
Politician	1,0000	0,9893	0,9946
PublicTransitSystem	0,9935	0,9776	0,9855
Ship	1,0000	0,9231	0,9600
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,9796	0,9658	0,9726
SportsEvent	1,0000	0,8636	0,9268
SportsLeague	0,9698	0,9835	0,9766
SportsManager	1,0000	0,9726	0,9861
SportsTeam	1,0000	0,9851	0,9925
Train	1,0000	1,0000	1,0000
Totals	**0,9870**	**0,9709**	**0,9789**

Table 3.76: Outcomes of global model in fine grained run with 400 abstracts from every domain

Description of the experiment: In this experiment we use the same trained model from previous experiment, but now the test file contains only abstracts from "POLITICS" domain, who is annotated in fine grain.

Results of the experiment: Table 3.77 show the output of the experiment, where we can see that even we have Election type on model and test file, the model do not find any entity with that type. Also for the Politician type we have a very low recall, which reflects that there is a very low overall result. Here we can also see which types the model recognize from other 2 domains, who are not part of the dataset. This also contributes to lower result.

In comparing with the experiment in coarse grain (see Table 3.73, now we have a bit better overall result. As well the results here are better than in main experiment from Table 3.7, even the model now recognize more types than there.

Entity	P	R	F1
Aircraft	0,0000	1,0000	0,0000
Election	0,0000	0,0000	0,0000
PoliticalParty	0,9766	0,9484	0,9623
Politician	1,0000	0,2092	0,3460
PublicTransitSystem	0,0000	1,0000	0,0000
Ship	0,0000	1,0000	0,0000
SportsClub	0,0000	1,0000	0,0000
SportsLeague	0,0000	1,0000	0,0000
Totals	**0,9619**	**0,4151**	**0,5799**

Table 3.77: Outcomes of global model in fine grained run with 400 abstracts from ”POLITICS” domain

Description of the experiment: How in the previous experiment also here we have the same model but now tested with abstracts from ”SPORT” domain annotated in fine grain.

Results of the experiment: In Table 3.78 we have the output of the provided experiment. How we can see the results are not bad at all for such big data. Comparable with the coarse grain model in Table 3.74 now we have a better overall result, although model also recognize wrong entities. Referencing to main experiment in Table 3.8 the results there are bit lower than now, even though than model recognize more wrong entities than in main experiment.

Entity	P	R	F1
Aircraft	0,0000	1,0000	0,0000
Athlete	1,0000	0,9899	0,9949
Coach	1,0000	1,0000	1,0000
PoliticalParty	0,0000	1,0000	0,0000
Politician	0,0000	1,0000	0,0000
SportsClub	0,9794	0,9654	0,9724
SportsEvent	1,0000	0,8636	0,9268
SportsLeague	0,9696	0,9834	0,9765
SportsManager	1,0000	0,9726	0,9861
SportsTeam	1,0000	0,9850	0,9924
Train	0,0000	1,0000	0,0000
Totals	**0,9821**	**0,9721**	**0,9770**

Table 3.78: Outcomes of global model in fine grained run with 400 abstracts from ”SPORT” domain

Description of the experiment: The last experiment with the model used in previous 3 experiment is now the dataset test file with abstracts from

"TRANSPORTATION" domain also annotated in fine grain.

Results of the experiment: Table 3.79 shows that our train model provide a bit better results than the coarse grain experiment in Table 3.75. Also comparing with the main experiment from Table 3.9, results there are little bit better than now. Than can be because model recognize one more wrong entity (PoliticalParty entity).

Entity	**P**	**R**	**F1**
Aircraft	1,0000	1,0000	1,0000
Automobile	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9896	0,9948
PoliticalParty	0,0000	1,0000	0,0000
Politician	0,0000	1,0000	0,0000
PublicTransitSystem	0,9934	0,9773	0,9853
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,0000	1,0000	0,0000
SportsTeam	0,0000	1,0000	0,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9866**	**0,9866**	**0,9866**

Table 3.79: Outcomes of global model in fine grained run with 400 abstracts from "TRANSPORTATION" domain

3.3.3.2 Evaluation of domain specific models with 400 abstracts

Description of the experiment: Experiment in Table 3.80 was provided with model trained only with abstracts from "POLITICS" domain in coarse grain. To train this model we needed 125.6 seconds, from which 114.88 seconds spent in optimization. The model is tested with the same dataset that is created.

Results of the experiment: How we can see from Table 3.80 the result is not bad at all for this big model. In comparing with experiments provided with global domain in Table 3.72 and in Table 3.73, the result now is better and a significant difference in result we can see in experiment where we had a global model tested with same dataset like here (see Table 3.73). Also referencing to main experiment from Table 3.10 where the only difference is the number of abstracts used for training the model, now the result is a better than there, although that now we have more data.

Entity	P	R	F1
POLITICS	0,9866	0,9479	0,9669
Totals	**0,9866**	**0,9479**	**0,9669**

Table 3.80: Outcome of ”POLITICS” domain specific model in coarse grained run with 400 abstracts from the same domain

Description of the experiment: For the purposes of this experiment we have used the same data from the previous one, but now annotated in fine grain. To train this kind of model we needed 416.7 seconds, from which 405.16 seconds spent in optimization.

Results of the experiment: We tested the model with the same data that was created and how we can see from Table 3.81 the overall result is better than previous experiment with coarse grain model. Also in comparing with the main experiment from Table 3.11 now model provides also a better result, but not that significant like in experiment from Table 3.77.

Entity	P	R	F1
Election	0,9975	0,9590	0,9779
PoliticalParty	0,9767	0,9530	0,9647
Politician	0,9977	0,9920	0,9948
Totals	**0,9906**	**0,9717**	**0,9810**

Table 3.81: Outcome of ”POLITICS” domain specific model in fine grained run with 400 abstracts from the same domain

Description of the experiment: Experiment in Table 3.82 is provided with a coarse grain model trained with abstracts only from ”SPORT” domain. The time needed to train this model was 90.9 seconds, from which 81.58 seconds spent in optimization. To test it we have used the same dataset that model was trained.

Results of the experiment: How we can see from Table 3.82 the trained model give a worthy results, who in comparing with the experiments provided with global model in Table 3.72 and Table 3.74 are a bit better. As well comparing with the results from the main experiment (see Table 3.12) we see that now results are also quite better, although we have a more entities.

Entity	P	R	F1
SPORT	0,9858	0,9676	0,9766
Totals	**0,9858**	**0,8676**	**0,9766**

Table 3.82: Outcome of ”SPORT” domain specific model in coarse grained run with 400 abstracts from the same domain

Description of the experiment: In this experiment we have used the same dataset from previous, but now entities are annotated in fine grain. To train a fine grain model we have need 915.1 seconds, from which 898.50 second spent in optimization. As well as previous the model is tested with the dataset that is trained.

Results of the experiment: In Table 3.83 we see the output of the experiment. Model give maximum precision on most of entities, as well the recall values are not bad at all but only one entity has maximum result. This results with a bit lower F1 score than the maximum value. In comparing with the previous experiment now overall result is a little bit better. Also comparing with the experiment with a global model in Table 3.78, again results are better. Final comparison is with main experiment in Table 3.13, where as well the result is significantly better now.

Entity	**P**	**R**	**F1**
Athlete	1,0000	0,9731	0,9863
Coach	1,0000	1,0000	1,0000
SportsClub	0,9815	0,9715	0,9765
SportsEvent	1,0000	0,9091	0,9524
SportsLeague	0,9718	0,9787	0,9752
SportsManager	1,0000	0,9726	0,9850
SportsTeam	1,0000	0,9900	0,9796
Totals	**0,9865**	**0,9727**	**0,9796**

Table 3.83: Outcome of "SPORT" domain specific model in fine grained run with 400 abstracts from the same domain

Description of the experiment: At the end we train a model with abstracts from "TRANSPORTATION" domain. To train a coarse grain model from this domain we needed 72.5 seconds, from which 64.53 seconds spent in optimization. Of coarse, here also we tested the model with the same dataset that was created.

Results of the experiment: In Table 3.84 we see that the model, unlike in previous experiment, gives values from every measurement close to maximum value. When we compare the results with the results from experiments provided with the global model in Table 3.72 and Table 3.75 we can see that now we again have a bit better result than in those experiments. Referring to main experiment in Table 3.14, the results are as well significantly better and more usable.

Entity	P	R	F1
TRANSPORTATION	0,9954	0,9747	0,9850
Totals	**0,9954**	**0,9747**	**0,9850**

Table 3.84: Outcome of ”TRANSPORTATION” domain specific model in coarse grained run with 400 abstracts from the same domain

Description of the experiment: We also train a fine grain model from ”TRANSPORTATION” domain. To train it we needed 725.2 seconds, from which 711.78 seconds spent in optimization. As well as the previous experiment, the model is tested with the same dataset that is created.

Results of the experiment: From the output of the experiment in Table 3.85 we can see that model provides excellent results, despite that, that is tested with big dataset. Comparing with the experiment who was provided with the global model and tested with same dataset like here in Table 3.79, now we have a bit better results, but a significant improvement on the results we have when we refer to the main experiment from Table 3.15.

Entity	P	R	F1
Aircraft	1,0000	0,9835	0,9917
Automobile	1,0000	0,9583	0,9787
Infrastructure	1,0000	0,9948	0,9974
PublicTransitSystem	0,9934	0,9773	0,9853
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	0,6667	0,8000
SpaceStation	1,0000	1,0000	1,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9970**	**0,9807**	**0,9888**

Table 3.85: Outcome of ”TRANSPORTATION” domain specific model in fine grained run with 400 abstracts from the same domain

3.3.3.3 Evaluation of global domain with 500 abstracts from every domain

Datasets with 500 abstracts for every domain. In this final group of experiments with 500 abstracts per domain we will repeat all experiments unlike in previous groups. To retrieve and prepare datasets for training in Stanford NER, our algorithm needs in total 3.63 hours.

Description of the experiment: In Table 3.86 we provide an experiment where the model is trained with, in total, 1500 abstracts annotated in coarse grain. To train this kind of model we needed 1021.3 seconds, from which 989.89 seconds spent in optimization. The model is tested with the dataset that was trained.

Results of the experiment: As we can see from Table 3.86 model provides amazing results even though that we have now a lot of data. From here we can see that results are closer to the main experiment from Table 3.2, but results there are bit better than now.

Entity	P	R	F1
POLITICS	0,9788	0,9444	0,9613
SPORT	0,9850	0,9596	0,9721
TRANSPORTATION	0,9962	0,9750	0,9855
Totals	**0,9857**	**0,9587**	**0,9720**

Table 3.86: Outcomes of global model in coarse grained run with 500 abstracts from every domain

Description of the experiment: For purposes of this experiment we have used the model train previously, but now it is tested with the dataset that contains only abstracts from ”POLITICS” domain, annotated in coarse grain.

Results of the experiment: From Table 3.87 we see that even the precision is not that bad, but because of very low recall, the F1 score is lower, which means that model recognize only half of our entities. Also model recognize entities that are not part of the tested dataset. As well when we compare with the previous experiment is clearly that there the results for ”POLITICS” domain are way more better. Also comparably with the main experiment from Table 3.3 results now are bit better, although there model recognize only some entities from ”TRANSPORTATION” domain.

Entity	P	R	F1
POLITICS	0,9734	0,4095	0,5765
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9549**	**0,4095**	**0,5732**

Table 3.87: Outcomes of global model in coarse grained run with 500 abstracts from ”POLITICS” domain

Description of the experiment: In this experiment we also used the global trained model, but now it is tested with ”SPORT” domain abstracts dataset.

Results of the experiment: Table 3.88 shows the output of the experiment, from where we see that even the results aren’t at their maximum are quite satisfying for such a big model and dataset. Also here model recognize entities that are not part of the dataset. As well in comparing with the results for ”SPORT” type in global model, now we have a very very little lower result. The same situation is with the main experiment in Table 3.4, where again results there are bit better.tter now.

Entity	P	R	F1
POLITICS	0,0000	1,0000	0,0000
SPORT	0,9849	0,9594	0,9720
TRANSPORTATION	0,0000	1,0000	0,0000
Totals	**0,9788**	**0,9594**	**0,9690**

Table 3.88: Outcomes of global model in coarse grained run with 500 abstracts from ”SPORT” domain

Description of the experiment: Final experiment with the global domain is that is tested with the dataset that contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: From Table 3.89 we see that model again recognize some entities from other 2 domains, that are not part of the tested dataset. Also if we compare only results for ”TRANSPORTATION” domain with the results from global experiment in Table 3.86, they are almost the same without any significant difference. The situation in main experiment from Table 3.5 is a little bit different, because there model recognize only one wrong entity and the results are bit better.

Entity	P	R	F1
POLITICS	0,0000	1,0000	0,0000
SPORT	0,0000	1,0000	0,0000
TRANSPORTATION	0,9961	0,9769	0,9864
Totals	**0,9870**	**0,9769**	**0,9819**

Table 3.89: Outcomes of global model in coarse grained run with 500 abstracts from ”TRANSPORTATION” domain

Description of the experiment: For purposes of this and the next 3 experiments we train a fine grain global model with 500 abstracts from every domain. To train this kind of model we needed 6706.7 seconds, from which 6650.30 seconds spent in optimization. In this experiment we tested the model with the dataset that was trained.

Results of the experiment: From Table 3.90 we can see the results for every particular ontology type. So even we have a lot of entities we see that results are not bad at all, even model provides maximum precision on most of entities and maximum recall on some of entities, which for a big model is excellent. When we compare with the results from coarse grain model, now we have a bit better overall result. As well referencing to main experiment in Table 3.6 the results there are bit worst than here, although here we have more data.

Entity	P	R	F1
Aircraft	1,0000	0,9929	0,9964
Athlete	1,0000	0,9896	0,9948
Automobile	1,0000	1,0000	1,0000
Coach	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9783	0,9890
PoliticalParty	0,9775	0,9403	0,9585
Politician	1,0000	0,9874	0,9937
PublicTransitSystem	0,9944	0,9807	0,9875
Ship	1,0000	0,9259	0,9615
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,9756	0,9553	0,9654
SportsEvent	1,0000	0,8796	0,9360
SportsLeague	0,9700	0,9810	0,9755
SportsManager	1,0000	0,9780	0,9889
SportsTeam	1,0000	0,9831	0,9915
Train	1,0000	1,0000	1,0000
Totals	**0,9866**	**0,9669**	**0,9766**

Table 3.90: Outcomes of global model in fine grained run with 500 abstracts from every domain

Description of the experiment: In this experiment we tested the global model with dataset that contains only abstracts from "POLITICS" domain, of coarse annotated in fine grain.

Results of the experiment: Table 3.91 shows the output of the experiment, where model provides terrible results and also recognize entity types which is not part of the dataset, but don't recognize any entity with Election type who is part of the dataset. This model for consolation gives a bit better results than the experiment with coarse grain model in Table 3.87. As well referring to main experiment in Table 3.7, now model gives a bit better results, but when we take also the size of the model this is a huge difference.

Entity	P	R	F1
Aircraft	0,0000	1,0000	0,0000
Election	0,0000	0,0000	0,0000
PoliticalParty	0,9774	0,9400	0,9583
Politician	1,0000	0,2171	0,3567
PublicTransitSystem	0,0000	1,0000	0,0000
Ship	0,0000	1,0000	0,0000
SportsClub	0,0000	1,0000	0,0000
SportsLeague	0,0000	1,0000	0,0000
Totals	**0,9631**	**0,4138**	**0,5789**

Table 3.91: Outcomes of global model in fine grained run with 500 abstracts from ”POLITICS” domain

Description of the experiment: Here we also have the global train model, but for this experiment is tested with dataset abstracts from ”SPORT” domain.

Results of the experiment: As we see from Table 3.92 model gives slightly better results than in the experiment with coarse grain model from Table 3.86. As well referencing to main experiment in Table 3.8, now model provides a bit lower result.

Entity	P	R	F1
Aircraft	0,0000	1,0000	0,0000
Athlete	1,0000	0,9896	0,9948
Coach	1,0000	1,0000	1,0000
PoliticalParty	0,0000	1,0000	0,0000
Politician	0,0000	1,0000	0,0000
SportsClub	0,9753	0,9549	0,9650
SportsEvent	1,0000	0,8796	0,9360
SportsLeague	0,9717	0,9810	0,9763
SportsManager	1,0000	0,9780	0,9889
SportsTeam	1,0000	0,9831	0,9915
Train	0,0000	1,0000	0,0000
Totals	**0,9785**	**0,9690**	**0,9737**

Table 3.92: Outcomes of global model in fine grained run with 500 abstracts from ”SPORT” domain

Description of the experiment: The final experiment with the global model is with dataset that contains only abstracts from ”TRANSPORTATION” domain.

Results of the experiment: Table 3.93 shows the outcome of the experiment, where for the types from ”TRANSPORTATION” domain we have an excellent results, but because model also recognize wrong entities, the overall results is lower. But in comparing with the experiment with coarse grain model in Table 3.89, now we have a bit better results. As well referring to main experiment in Table 3.9, we now have again a bit lower result.

Entity	**P**	**R**	**F1**
Aircraft	1,0000	0,9927	0,9963
Automobile	1,0000	1,0000	1,0000
Infrastructure	1,0000	0,9783	0,9890
PoliticalParty	0,0000	1,0000	0,0000
Politician	0,0000	1,0000	0,0000
PublicTransitSystem	0,9943	0,9804	0,9873
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	1,0000	1,0000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,0000	1,0000	0,0000
SportsTeam	0,0000	1,0000	0,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9884**	**0,9833**	**0,9858**

Table 3.93: Outcomes of global model in fine grained run with 500 abstracts from "TRANSPORTATION" domain

3.3.3.4 Evaluation of domain specific models with 500 abstracts

Description of the experiment: In this experiment we train a domain specific coarse grain model with dataset that contains only abstracts from "POLITICS" domain. The time needed to train this model was 165.6 seconds, from which 152.45 seconds spent in optimization. The model is tested with the same dataset that is trained.

Results of the experiment: As we can see from Table 3.94 this model provides a better result than the 2 experiments with the global model in Table 3.86 and and a way more better results than experiment from Table 3.87. As well referencing to main experiment in Table 3.10, now model gives better results.

Entity	**P**	**R**	**F1**
POLITICS	0,9808	0,9450	0,9626
Totals	**0,9808**	**0,9450**	**0,9626**

Table 3.94: Outcome of "POLITICS" domain specific model in coarse grained run with 500 abstracts from the same domain

textbfDescription of the experiment: With the same data from the previous experiment, but not annotated in fine grain, we train new model. We needed 479.3 seconds to train this model, from which 465.16 seconds spent in optimization. The model is tested with the same dataset that was trained.

Results of the experiment: Table 3.95 shows the outcome of the experiment, where we see that even results on every type are close to maximum, the overall results is a bit better than the previous experiment. But in comparing to the experiment with global domain inTable 3.91, now results are way more better, we don't have any wrong recognition and also entities from Election ontology type are recognized. Referencing to the main experiment in Table 3.11, as well now model provides a better results, which is surprisingly for such big dataset.

Entity	**P**	**R**	**F1**
Election	0,9915	0,9393	0,9647
PoliticalParty	0,9777	0,9502	0,9637
Politician	0,9962	0,9877	0,9919
Totals	**0,9890**	**0,9648**	**0,9768**

Table 3.95: Outcome of "POLITICS" domain specific model in fine grained run with 500 abstracts from the same domain

Description of the experiment: For the purposes of the experiment we train a coarse grain model with dataset that contains only abstracts from "SPORT" domain. Time needed to train this kind of model was 115.0 seconds, from which 103.59 seconds spent in optimization. The model is tested like in previous experiments, with the same dataset that is trained.

Results of the experiment: From Table 3.96 we see that model gives a slightly better results than in experiments with global domain in Table 3.86 and Table 3.88. Also referencing to the main experiment in Table 3.12, now model gives again significant better results than there.

Entity	**P**	**R**	**F1**
SPORT	0,9856	0,9706	0,9780
Totals	**0,9856**	**0,9706**	**0,9780**

Table 3.96: Outcome of "SPORT" domain specific model in coarse grained run with 500 abstracts from the same domain

Description of the experiment: This experiment is provided with a "SPORT" fine grain model. To train this model we needed 1175.1 seconds, from which 1158.90 seconds spent in optimization. Test is the same like previous experiment, where dataset now are annotated in fine grain.

Results of the experiment: As we see from Table 3.92 the overall results is a bit better than previous experiment, as well better than experiment provided with the global model in Table 3.92. Also referencing to main experiment in Table 3.13, now model was more precise than there, an because of that gives better results.

Entity	**P**	**R**	**F1**
Athlete	1,0000	0,9791	0,9894
Coach	1,0000	1,0000	1,0000
SportsClub	0,9771	0,9630	0,9700
SportsEvent	1,0000	0,9074	0,9515
SportsLeague	0,9755	0,9848	0,9801
SportsManager	1,0000	0,9780	0,9889
SportsTeam	1,0000	0,9915	0,9957
Totals	**0,9861**	**0,9731**	**0,9796**

Table 3.97: Outcome of ”SPORT” domain specific model in fine grained run with 500 abstracts from the same domain

Description of the experiment: Experiment in Table 3.98 is provided with a coarse grain model train with dataset from ”TRANSPORTATION” domain. To train this model we needed 78.3 seconds in total, from which 67.96 seconds spent in optimization. As previous model is tested with same dataset that is trained.

Results of the experiment: Table 3.98 shows that model provides very good results on every measurement. Comparably with experiments with global model in Table 3.86 and Table 3.89, now model gives better overall score than there and it's faster to train and provide experiment. As well referencing to main experiment in Table 3.14 again results from this experiment are significantly better than main experiment results.

Entity	**P**	**R**	**F1**
TRANSPORTATION	0,9974	0,9756	0,9864
Totals	**0,9974**	**0,9756**	**0,9864**

Table 3.98: Outcome of ”TRANSPORTATION” domain specific model in coarse grained run with 500 abstracts from the same domain

Description of the experiment: The final experiment with this group of number of abstracts is fine grain model for ”TRANSPORTATION” domain. Time taken to train this kind of model was in total 1040.2 seconds, from which 1023.55 seconds, spent in optimization. Testing routine is the same here.

Results of the experiment: From Table 3.99 we again have excellent results. Comparing with the experiment provided with global model in Table 3.93 and the main experiment in Table 3.15, model here gives a better results, than in those experiments.

Entity	P	R	F1
Aircraft	1,0000	0,9781	0,9889
Automobile	1,0000	0,8800	0,9362
Infrastructure	1,0000	0,9870	0,9934
PublicTransitSystem	0,9915	0,9804	0,9860
Ship	1,0000	1,0000	1,0000
SpaceShuttle	1,0000	0,6667	0,8000
SpaceStation	1,0000	1,0000	1,0000
Train	1,0000	1,0000	1,0000
Totals	**0,9961**	**0,9769**	**0,9864**

Table 3.99: Outcome of ”TRANSPORTATION” domain specific model in coarse grained run with 500 abstracts from the same domain

CONLUSION!!!!!!!!!!!!!!!!!!!!!!!

3.3.4 MIXED

Entity	**P**	**R**	**F1**
Aircraft	0,9242	0,5755	0,7093
Athlete	0,8182	0,3778	0,5169
Automobile	0,9565	0,3607	0,5238
Coach	1,0000	0,2000	0,3333
Infrastructure	1,0000	0,9896	0,9948
Locomotive	0,0000	0,0000	0,0000
Motorcycle	0,0000	0,0000	0,0000
OrganisationMember	0,0000	0,0000	0,0000
PoliticalParty	0,7656	0,5819	0,6613
Politician	0,8925	0,4099	0,5618
PublicTransitSystem	0,8291	0,6178	0,7080
Ship	0,9375	0,3409	0,5000
SpaceShuttle	1,0000	0,3750	0,5455
SpaceStation	1,0000	0,3333	0,5000
SportsClub	0,8009	0,4276	0,5575
SportsEvent	0,9559	0,3171	0,4762
SportsLeague	0,8071	0,5912	0,6824
SportsManager	0,9643	0,2903	0,4463
SportsTeam	0,8856	0,5838	0,7037
Train	1,0000	0,5455	0,7059
Totals	**0,8206**	**0,4837**	**0,6087**

Table 3.100: All 3 Domains Fine Grained Top 300 With All 3 Domains Fine Grained Top 500 Links And All 3 Domains Fine Grained Top 500 Links With Lower PageRank

Entity	**P**	**R**	**F1**
Aircraft	0,9735	0,6934	0,8099
Athlete	0,9101	0,6222	0,7391
Automobile	1,0000	0,4098	0,5814
Coach	1,0000	0,3000	0,4615
Infrastructure	0,8885	0,5218	0,6575
Locomotive	0,0000	0,0000	0,0000
Motorcycle	0,0000	0,0000	0,0000
OrganisationMember	0,0000	0,0000	0,0000
PoliticalParty	0,8393	0,7403	0,7876
Politician	0,9271	0,6593	0,7706
PublicTransitSystem	0,9027	0,7389	0,8126
Rocket	0,0000	0,0000	0,0000
Ship	0,9615	0,5682	0,7143
SpaceShuttle	1,0000	0,4375	0,6087
SpaceStation	1,0000	0,6667	0,8000
SportsClub	0,8722	0,6071	0,7159
SportsEvent	0,9000	0,4829	0,6286
SportsLeague	0,8622	0,7357	0,7939
SportsManager	0,9787	0,4946	0,6571
SportsTeam	0,9276	0,7514	0,8302
Train	1,0000	0,5455	0,7059
Totals	**0,8844**	**0,6592**	**0,7553**

Table 3.101: All 3 Domains Fine Grained Top 500 Links With All 3 Domains Fine Grained Top 500 Links And All 3 Domains Fine Grained Top 500 Links With Lower Page Rank

Entity	P	R	F1
Aircraft	0,6667	0,1111	0,1905
Athlete	0,4675	0,1268	0,1994
Automobile	0,0000	0,0000	0,0000
Coach	0,0000	0,0000	0,0000
Infrastructure	0,5352	0,1387	0,2203
Locomotive	0,0000	0,0000	0,0000
Motorcycle	0,0000	0,0000	0,0000
OrganisationMember	0,0000	0,0000	0,0000
PoliticalParty	0,5462	0,4097	0,4682
Politician	0,5962	0,1867	0,2844
PublicTransitSystem	0,6943	0,4098	0,5154
Rocket	0,0000	0,0000	0,0000
Ship	0,0000	0,0000	0,0000
SpaceShuttle	0,0000	0,0000	0,0000
SpaceStation	0,0000	0,0000	0,0000
SportsClub	0,6370	0,2675	0,3768
SportsEvent	0,2667	0,0412	0,0714
SportsLeague	0,6395	0,4125	0,5015
SportsManager	0,6000	0,0316	0,0600
SportsTeam	0,6316	0,2975	0,4045
Train	0,0000	0,0000	0,0000
Totals	**0,5983**	**0,2670**	**0,3692**

Table 3.102: All 3 Domains Fine Grained Top 500 Links With All 3 Domains Fine Grained Top 500 Links With Lower PageRank

Entity	P	R	F1
Aircraft	0,9950	0,9706	0,9826
Athlete	0,0000	0,0000	0,0000
Automobile	1,0000	0,8500	0,9189
Coach	0,0000	0,0000	0,0000
Infrastructure	1,0000	0,9820	0,9909
PoliticalParty	0,0000	0,0000	0,0000
Politician	0,0000	0,0000	0,0000
PublicTransitSystem	0,9835	0,9676	0,9755
Ship	1,0000	0,9655	0,9825
SpaceShuttle	1,0000	0,6667	0,8000
SpaceStation	1,0000	1,0000	1,0000
SportsClub	0,0000	0,0000	0,0000
SportsEvent	0,0000	0,0000	0,0000
SportsLeague	0,0000	0,0000	0,0000
SportsManager	0,0000	0,0000	0,0000
SportsTeam	0,0000	0,0000	0,0000
Train	1,0000	0,9091	0,9524
Totals	**0,9909**	**0,3491**	**0,5163**

Table 3.103: Transportation Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 300 Links And Transportation Fine Grained Top 300 Links

3.3.5 Experiments with lower training abstracts on model, but higher abstracts on test file

```
CRFClassifier tagged 97089 words in 1 documents at 2905,12 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  0,0435  0,0833  1       0       22
        Athlete 1,0000  0,0278  0,0541  1       0       35
     Automobile 0,0000  0,0000  0,0000  0       0       3
          Coach 1,0000  0,3333  0,5000  1       0       2
 Infrastructure 0,0000  0,0000  0,0000  0       0       22
 PoliticalParty 0,6458  0,3054  0,4147  62      34      141
     Politician 1,0000  0,0727  0,1356  4       0       51
PublicTransitSystem     0,8750  0,1346  0,2333  7       1       45
           Ship 1,0000  0,2000  0,3333  1       0       4
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,9000  0,1084  0,1935  9       1       74
    SportsEvent 1,0000  0,0233  0,0455  1       0       42
   SportsLeague 0,6667  0,2121  0,3218  14      7       52
  SportsManager 0,0000  0,0000  0,0000  0       0       6
     SportsTeam 1,0000  0,0303  0,0588  1       0       32
          Train 0,0000  0,0000  0,0000  0       0       1
         Totals 0,7034  0,1591  0,2595  102     43      539

C:\Dev\stanford-ner-2017-06-09>java -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier modelsWit
hOneAnnotation\Top10Links\ner-All3DomainsTop10LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100Links\All
3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

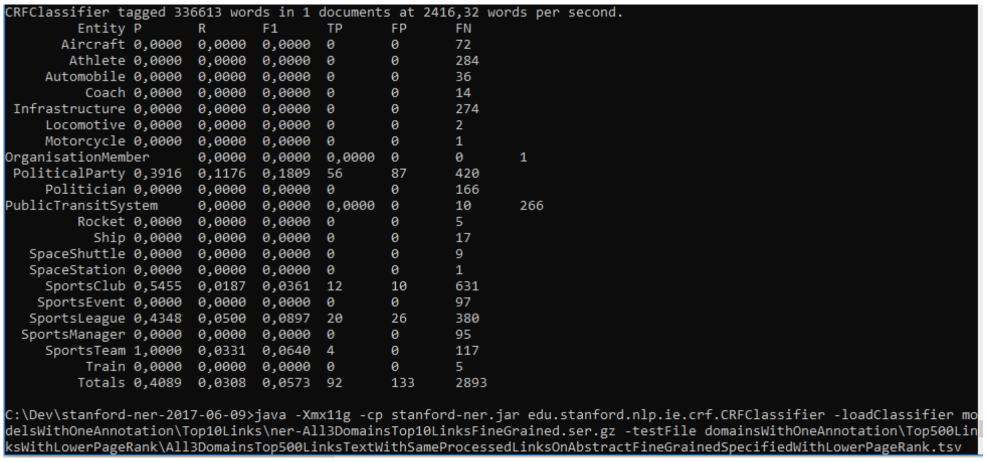
Figure 3.1: All 3 Domains Fine Grained Top 10 Links Runned With All 3 Domains Fine Grained Top 100 Links

```
CRFClassifier tagged 270361 words in 1 documents at 3542,70 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  0,0098  0,0194  1       0       101
        Athlete 1,0000  0,0050  0,0099  1       0       201
     Automobile 0,0000  0,0000  0,0000  0       0       20
          Coach 1,0000  0,2500  0,4000  1       0       3
 Infrastructure 0,0000  0,0000  0,0000  0       0       111
 PoliticalParty 0,5517  0,2192  0,3137  112     91      399
     Politician 0,8000  0,0288  0,0556  4       1       135
PublicTransitSystem     0,6364  0,0282  0,0541  7       4       241
           Ship 1,0000  0,0667  0,1250  1       0       14
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,8750  0,0403  0,0771  14      2       333
    SportsEvent 1,0000  0,0152  0,0299  1       0       65
   SportsLeague 0,6102  0,1173  0,1967  36      23      271
  SportsManager 0,0000  0,0000  0,0000  0       0       52
     SportsTeam 1,0000  0,0065  0,0129  1       0       153
          Train 0,0000  0,0000  0,0000  0       0       6
         Totals 0,5967  0,0781  0,1382  179     121     2112

C:\Dev\stanford-ner-2017-06-09>java -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier modelsWit
hOneAnnotation\Top10Links\ner-All3DomainsTop10LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300Links\All
3DomainsTop300LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.2: All 3 Domains Fine Grained Top 10 Links Runned With All 3 Domains Fine Grained Top 300 Links

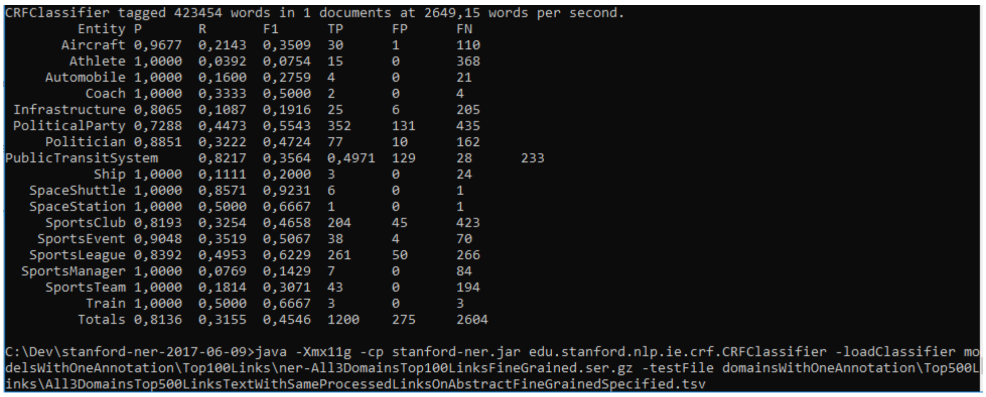
```
CRFClassifier tagged 423454 words in 1 documents at 3197,54 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  0,0071  0,0142  1       0       139
        Athlete 1,0000  0,0026  0,0052  1       0       382
     Automobile 0,0000  0,0000  0,0000  0       0       25
          Coach 1,0000  0,1667  0,2857  1       0       5
 Infrastructure 0,0000  0,0000  0,0000  0       0       230
 PoliticalParty 0,5385  0,1957  0,2870  154     132     633
     Politician 0,8000  0,0167  0,0328  4       1       235
PublicTransitSystem     0,4118  0,0193  0,0369  7       10      355
           Ship 1,0000  0,0370  0,0714  1       0       26
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       7
   SpaceStation 0,0000  0,0000  0,0000  0       0       2
     SportsClub 0,8148  0,0351  0,0673  22      5       605
    SportsEvent 1,0000  0,0093  0,0183  1       0       107
   SportsLeague 0,5556  0,0854  0,1480  45      36      482
  SportsManager 0,0000  0,0000  0,0000  0       0       91
     SportsTeam 1,0000  0,0295  0,0574  7       0       230
          Train 0,0000  0,0000  0,0000  0       0       6
         Totals 0,5701  0,0641  0,1153  244     184     3560

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier m
odelsWithOneAnnotation\Top10Links\ner-All3DomainsTop10LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.3: All 3 Domains Fine Grained Top 10 Links Runned With All 3 Domains Fine Grained Top 500 Links



```
CRFClassifier tagged 336613 words in 1 documents at 2416,32 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       72
        Athlete 0,0000  0,0000  0,0000  0       0       284
     Automobile 0,0000  0,0000  0,0000  0       0       36
          Coach 0,0000  0,0000  0,0000  0       0       14
 Infrastructure 0,0000  0,0000  0,0000  0       0       274
     Locomotive 0,0000  0,0000  0,0000  0       0       2
     Motorcycle 0,0000  0,0000  0,0000  0       0       1
OrganisationMember      0,0000  0,0000  0,0000  0       0       1
 PoliticalParty 0,3916  0,1176  0,1809  56      87      420
     Politician 0,0000  0,0000  0,0000  0       0       166
PublicTransitSystem     0,0000  0,0000  0,0000  0       10      266
         Rocket 0,0000  0,0000  0,0000  0       0       5
           Ship 0,0000  0,0000  0,0000  0       0       17
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       9
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,5455  0,0187  0,0361  12      10      631
    SportsEvent 0,0000  0,0000  0,0000  0       0       97
   SportsLeague 0,4348  0,0500  0,0897  20      26      380
  SportsManager 0,0000  0,0000  0,0000  0       0       95
     SportsTeam 1,0000  0,0331  0,0640  4       0       117
          Train 0,0000  0,0000  0,0000  0       0       5
         Totals 0,4089  0,0308  0,0573  92      133     2893

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top10Links\ner-All3DomainsTop10LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500Lin
ksWithLowerPageRank\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecifiedWithLowerPageRank.tsv
```

Figure 3.4: All 3 Domains Fine Grained Top 10 Links Runned With All 3 Domains Fine Grained Top 500 Links With Lower PageRank



```
CRFClassifier tagged 423454 words in 1 documents at 2649,15 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,9677  0,2143  0,3509  30      1       110
        Athlete 1,0000  0,0392  0,0754  15      0       368
     Automobile 1,0000  0,1600  0,2759  4       0       21
          Coach 1,0000  0,3333  0,5000  2       0       4
 Infrastructure 0,8065  0,1087  0,1916  25      6       205
 PoliticalParty 0,7288  0,4473  0,5543  352     131     435
     Politician 0,8851  0,3222  0,4724  77      10      162
PublicTransitSystem     0,8217  0,3564  0,4971  129     28      233
           Ship 1,0000  0,1111  0,2000  3       0       24
   SpaceShuttle 1,0000  0,8571  0,9231  6       0       1
   SpaceStation 1,0000  0,5000  0,6667  1       0       1
     SportsClub 0,8193  0,3254  0,4658  204     45      423
    SportsEvent 0,9048  0,3519  0,5067  38      4       70
   SportsLeague 0,8392  0,4953  0,6229  261     50      266
  SportsManager 1,0000  0,0769  0,1429  7       0       84
     SportsTeam 1,0000  0,1814  0,3071  43      0       194
          Train 1,0000  0,5000  0,6667  3       0       3
         Totals 0,8136  0,3155  0,4546  1200    275     2604

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.5: All 3 Domains Fine Grained Top 100 Links Runned With All 3 Domains Fine Grained Top 500 Links

```
CRFClassifier tagged 336613 words in 1 documents at 3519,92 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       1       72
        Athlete 0,0000  0,0000  0,0000  0       0       284
     Automobile 0,0000  0,0000  0,0000  0       0       36
          Coach 0,0000  0,0000  0,0000  0       0       14
 Infrastructure 0,0000  0,0000  0,0000  0       7       274
     Locomotive 0,0000  0,0000  0,0000  0       0       2
     Motorcycle 0,0000  0,0000  0,0000  0       0       1
OrganisationMember      0,0000  0,0000  0,0000  0       0       1
 PoliticalParty 0,5022  0,2374  0,3224  113     112     363
     Politician 0,5517  0,0964  0,1641  16      13      150
PublicTransitSystem     0,7255  0,1391  0,2334  37      14      229
         Rocket 0,0000  0,0000  0,0000  0       0       5
           Ship 0,0000  0,0000  0,0000  0       0       17
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       9
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,6093  0,1431  0,2317  92      59      551
    SportsEvent 0,1667  0,0103  0,0194  1       5       96
   SportsLeague 0,6092  0,2650  0,3693  106     68      294
  SportsManager 0,0000  0,0000  0,0000  0       1       95
     SportsTeam 0,8571  0,0496  0,0937  6       1       115
          Train 0,0000  0,0000  0,0000  0       0       5
         Totals 0,5690  0,1243  0,2040  371     281     2614

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecifiedWithLowerPageRank.tsv
```

Figure 3.6: All 3 Domains Fine Grained Top 100 Links Runned With All 3 Domains Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 164060 words in 1 documents at 3254,58 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       494
 PoliticalParty 0,7282  0,4483  0,5549  351     131     432
     Politician 0,9398  0,0739  0,1371  78      5       977
    SportsEvent 0,0000  1,0000  0,0000  0       1       0
   SportsLeague 0,0000  1,0000  0,0000  0       5       0
         Totals 0,7513  0,1840  0,2956  429     142     1903

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\PoliticsFineGrainedTop500Links.tsv
```

Figure 3.7: All 3 Domains Fine Grained Top 100 Links Runned With Politics Fine Grained Top 500 Links

```
CRFClassifier tagged 117053 words in 1 documents at 3034,66 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       197
 Infrastructure 0,0000  1,0000  0,0000  0       2       0
 PoliticalParty 0,5045  0,2379  0,3233  113     111     362
     Politician 0,6667  0,0245  0,0473  16      8       636
PublicTransitSystem     0,0000  1,0000  0,0000  0       3       0
    SportsEvent 0,0000  1,0000  0,0000  0       2       0
   SportsLeague 0,0000  1,0000  0,0000  0       3       0
         Totals 0,5000  0,0974  0,1631  129     129     1195

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\PoliticsFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.8: All 3 Domains Fine Grained Top 100 Links Runned With Politics Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 142322 words in 1 documents at 3035,30 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  0,0392  0,0754  15      0       368
          Coach 1,0000  0,3333  0,5000  2       0       4
 Infrastructure 0,0000  1,0000  0,0000  0       1       0
 PoliticalParty 0,0000  1,0000  0,0000  0       1       0
     Politician 0,0000  1,0000  0,0000  0       1       0
     SportsClub 0,8185  0,3269  0,4672  203     45      418
    SportsEvent 0,9268  0,3519  0,5101  38      3       70
   SportsLeague 0,8497  0,4952  0,6258  260     46      265
  SportsManager 1,0000  0,0769  0,1429  7       0       84
     SportsTeam 1,0000  0,1780  0,3022  42      0       194
         Totals 0,8539  0,2878  0,4305  567     97      1403

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\SportFineGrainedTop500Links.tsv
```

Figure 3.9: All 3 Domains Fine Grained Top 100 Links Runned With Sport Fine Grained Top 500 Links

```
CRFClassifier tagged 118469 words in 1 documents at 3761,28 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 0,0000  0,0000  0,0000  0       0       271
          Coach 0,0000  0,0000  0,0000  0       0       14
 Infrastructure 0,0000  1,0000  0,0000  0       1       0
OrganisationMember      0,0000  0,0000  0,0000  0       0       1
     Politician 0,0000  1,0000  0,0000  0       4       0
     SportsClub 0,6216  0,1467  0,2374  92      56      535
    SportsEvent 0,2500  0,0104  0,0200  1       3       95
   SportsLeague 0,6082  0,2620  0,3662  104     67      293
  SportsManager 0,0000  0,0000  0,0000  0       1       89
     SportsTeam 0,8571  0,0504  0,0952  6       1       113
         Totals 0,6042  0,1258  0,2082  203     133     1411

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\SportFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.10: All 3 Domains Fine Grained Top 100 Links Runned With Sport Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 117072 words in 1 documents at 3791,68 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,9677  0,2190  0,3571  30      1       107
     Automobile 1,0000  0,1600  0,2759  4       0       21
 Infrastructure 0,8333  0,1087  0,1923  25      5       205
     Politician 0,0000  1,0000  0,0000  0       3       0
PublicTransitSystem     0,8217  0,3603  0,5010  129     28      229
           Ship 1,0000  0,1875  0,3158  3       0       13
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  0,5000  0,6667  1       0       1
     SportsClub 0,0000  1,0000  0,0000  0       1       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
          Train 1,0000  0,6000  0,7500  3       0       2
         Totals 0,8375  0,2580  0,3945  201     39      578

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\TransportationFineGrainedTop500Links.tsv
```

Figure 3.11: All 3 Domains Fine Grained Top 100 Links Runned With Transportation Fine Grained Top 500 Links

```
CRFClassifier tagged 101091 words in 1 documents at 3831,82 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       1       72
     Automobile 0,0000  0,0000  0,0000  0       0       36
 Infrastructure 0,0000  0,0000  0,0000  0       4       272
     Locomotive 0,0000  0,0000  0,0000  0       0       2
     Motorcycle 0,0000  0,0000  0,0000  0       0       1
 PoliticalParty 0,0000  1,0000  0,0000  0       1       0
     Politician 0,0000  1,0000  0,0000  0       1       0
PublicTransitSystem     0,7083  0,1399  0,2337  34      14      209
         Rocket 0,0000  0,0000  0,0000  0       0       5
           Ship 0,0000  0,0000  0,0000  0       0       12
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       9
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  1,0000  0,0000  0       3       0
          Train 0,0000  0,0000  0,0000  0       0       5
         Totals 0,5862  0,0517  0,0950  34      24      624

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top100Links\ner-All3DomainsTop100LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\TransportationFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.12: All 3 Domains Fine Grained Top 100 Links Runned With Transportation Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 423454 words in 1 documents at 2349,18 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,9573  0,8000  0,8716  112     5       28
        Athlete 0,8889  0,5849  0,7055  224     28      159
     Automobile 0,9565  0,8800  0,9167  22      1       3
          Coach 1,0000  0,6667  0,8000  4       0       2
 Infrastructure 0,9034  0,5696  0,6987  131     14      99
 PoliticalParty 0,8854  0,7268  0,7983  572     74      215
     Politician 0,9664  0,6025  0,7423  144     5       95
PublicTransitSystem     0,8966  0,7901  0,8399  286     33      76
           Ship 1,0000  0,5556  0,7143  15      0       12
   SpaceShuttle 1,0000  0,8571  0,9231  6       0       1
   SpaceStation 1,0000  0,5000  0,6667  1       0       1
     SportsClub 0,9018  0,6443  0,7516  404     44      223
    SportsEvent 0,9688  0,5741  0,7209  62      2       46
   SportsLeague 0,9079  0,7856  0,8423  414     42      113
  SportsManager 0,9811  0,5714  0,7222  52      1       39
     SportsTeam 0,9784  0,7637  0,8578  181     4       56
          Train 1,0000  1,0000  1,0000  6       0       0
         Totals 0,9124  0,6930  0,7877  2636    253     1168

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.13: All 3 Domains Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 500 Links

```
CRFClassifier tagged 336613 words in 1 documents at 3545,42 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,6667  0,1389  0,2299  10      5       62
        Athlete 0,5000  0,0986  0,1647  28      28      256
     Automobile 0,0000  0,0000  0,0000  0       0       36
          Coach 0,0000  0,0000  0,0000  0       0       14
 Infrastructure 0,5667  0,0620  0,1118  17      13      257
     Locomotive 0,0000  0,0000  0,0000  0       0       2
     Motorcycle 0,0000  0,0000  0,0000  0       0       1
OrganisationMember      0,0000  0,0000  0,0000  0       0       1
 PoliticalParty 0,5191  0,3424  0,4127  163     151     313
     Politician 0,5946  0,1325  0,2167  22      15      144
PublicTransitSystem     0,6846  0,3835  0,4916  102     47      164
         Rocket 0,0000  0,0000  0,0000  0       0       5
           Ship 0,0000  0,0000  0,0000  0       1       17
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       9
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,6043  0,2162  0,3184  139     91      504
    SportsEvent 0,7500  0,0309  0,0594  3       1       94
   SportsLeague 0,6009  0,3350  0,4302  134     89      266
  SportsManager 0,6667  0,0211  0,0408  2       1       93
     SportsTeam 0,5490  0,2314  0,3256  28      23      93
          Train 0,0000  0,0000  0,0000  0       0       5
         Totals 0,5822  0,2171  0,3163  648     465     2337

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\All3DomainsTop500LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecifiedWithLowerPageRank.tsv
```

Figure 3.14: All 3 Domains Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 164060 words in 1 documents at 4066,23 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 0,0000  1,0000  0,0000  0       9       0
       Election 0,0000  0,0000  0,0000  0       0       494
 PoliticalParty 0,8853  0,7292  0,7997  571     74      212
     Politician 0,9792  0,1336  0,2352  141     3       914
PublicTransitSystem     0,0000  1,0000  0,0000  0       2       0
           Ship 0,0000  1,0000  0,0000  0       1       0
     SportsClub 0,0000  1,0000  0,0000  0       1       0
   SportsLeague 0,0000  1,0000  0,0000  0       2       0
  SportsManager 0,0000  1,0000  0,0000  0       1       0
         Totals 0,8845  0,3053  0,4539  712     93      1620

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\PoliticsFineGrainedTop500Links.tsv
```

Figure 3.15: All 3 Domains Fine Grained Top 300 Links Runned With Politics Fine Grained Top 500 Links

```
CRFClassifier tagged 117053 words in 1 documents at 4211,90 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 0,0000  1,0000  0,0000  0       6       0
       Election 0,0000  0,0000  0,0000  0       0       197
 Infrastructure 0,0000  1,0000  0,0000  0       1       0
 PoliticalParty 0,5209  0,3411  0,4122  162     149     313
     Politician 0,5882  0,0307  0,0583  20      14      632
PublicTransitSystem     0,0000  1,0000  0,0000  0       9       0
   SportsLeague 0,0000  1,0000  0,0000  0       2       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
         Totals 0,5000  0,1375  0,2156  182     182     1142

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\PoliticsFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.16: All 3 Domains Fine Grained Top 300 Links Runned With Politics Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 142322 words in 1 documents at 4466,12 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  1,0000  0,0000  0       1       0
        Athlete 0,9218  0,5849  0,7157  224     19      159
     Automobile 0,0000  1,0000  0,0000  0       1       0
          Coach 1,0000  0,6667  0,8000  4       0       2
 PoliticalParty 0,0000  1,0000  0,0000  0       1       0
     Politician 0,0000  1,0000  0,0000  0       3       0
     SportsClub 0,9009  0,6441  0,7512  400     44      221
    SportsEvent 0,9688  0,5741  0,7209  62      2       46
   SportsLeague 0,9097  0,7867  0,8437  413     41      112
  SportsManager 1,0000  0,5714  0,7273  52      0       39
     SportsTeam 0,9783  0,7627  0,8571  180     4       56
          Train 0,0000  1,0000  0,0000  0       1       0
         Totals 0,9194  0,6777  0,7802  1335    117     635

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\SportFineGrainedTop500Links.tsv
```

Figure 3.17: All 3 Domains Fine Grained Top 300 Links Runned With Sport Fine Grained Top 500 Links

```
CRFClassifier tagged 118469 words in 1 documents at 4461,27 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 0,5833  0,1033  0,1755  28      20      243
          Coach 0,0000  0,0000  0,0000  0       0       14
OrganisationMember      0,0000  0,0000  0,0000  0       0       1
 PoliticalParty 0,0000  1,0000  0,0000  0       3       0
     Politician 0,0000  1,0000  0,0000  0       3       0
     SportsClub 0,6096  0,2217  0,3251  139     89      488
    SportsEvent 0,7500  0,0313  0,0600  3       1       93
   SportsLeague 0,6000  0,3325  0,4279  132     88      265
  SportsManager 0,6667  0,0225  0,0435  2       1       87
     SportsTeam 0,5833  0,2353  0,3353  28      20      91
         Totals 0,5961  0,2057  0,3058  332     225     1282

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\SportFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.18: All 3 Domains Fine Grained Top 300 Links Runned With Sport Fine Grained Top 500 Links With Lower PageRank

```
CRFClassifier tagged 117072 words in 1 documents at 4190,42 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,9569  0,8102  0,8775  111     5       26
     Automobile 1,0000  0,8800  0,9362  22      0       3
 Infrastructure 0,9034  0,5696  0,6987  131     14      99
     Politician 0,0000  1,0000  0,0000  0       2       0
PublicTransitSystem     0,8959  0,7933  0,8415  284     33      74
           Ship 1,0000  0,8750  0,9333  14      0       2
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  0,5000  0,6667  1       0       1
     SportsClub 0,0000  1,0000  0,0000  0       3       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
          Train 1,0000  1,0000  1,0000  5       0       0
         Totals 0,9082  0,7368  0,8136  574     58      205

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inks\TransportationFineGrainedTop500Links.tsv
```

Figure 3.19: All 3 Domains Fine Grained Top 300 Links Runned With Transportation Fine Grained Top 500 Links

```
CRFClassifier tagged 101091 words in 1 documents at 3283,24 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,6667  0,1389  0,2299  10      5       62
        Athlete 0,0000  1,0000  0,0000  0       2       0
     Automobile 0,0000  0,0000  0,0000  0       0       36
 Infrastructure 0,5862  0,0625  0,1130  17      12      255
     Locomotive 0,0000  0,0000  0,0000  0       0       2
     Motorcycle 0,0000  0,0000  0,0000  0       0       1
PublicTransitSystem     0,6714  0,3868  0,4909  94      46      149
         Rocket 0,0000  0,0000  0,0000  0       0       5
           Ship 0,0000  0,0000  0,0000  0       1       12
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       9
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  1,0000  0,0000  0       2       0
   SportsLeague 0,0000  1,0000  0,0000  0       1       0
     SportsTeam 0,0000  1,0000  0,0000  0       2       0
          Train 0,0000  0,0000  0,0000  0       0       5
         Totals 0,6302  0,1839  0,2847  121     71      537

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top500L
inksWithLowerPageRank\TransportationFineGrainedTop500LinksWithLowerPageRank.tsv
```

Figure 3.20: All 3 Domains Fine Grained Top 300 Links Runned With Transportation Fine Grained Top 500 Links With Lower PageRank

3.3.6 Experiments with higher training abstracts on model, but lower abstracts on test file

```
CRFClassifier tagged 10336 words in 1 documents at 2532,71 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  1       0       0
        Athlete 1,0000  1,0000  1,0000  1       0       0
          Coach 1,0000  1,0000  1,0000  1       0       0
  PoliticalParty 0,9545  0,8077  0,8750  21      1       5
     Politician 1,0000  1,0000  1,0000  4       0       0
PublicTransitSystem     1,0000  0,8571  0,9231  6       0       1
           Ship 1,0000  1,0000  1,0000  1       0       0
     SportsClub 1,0000  1,0000  1,0000  7       0       0
    SportsEvent 1,0000  1,0000  1,0000  1       0       0
   SportsLeague 1,0000  0,7500  0,8571  3       0       1
     SportsTeam 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9792  0,8704  0,9216  47      1       7

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\All3DomainsTop10LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.21: All 3 Domains Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 10 Links

```
CRFClassifier tagged 97089 words in 1 documents at 4145,56 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  23      0       0
        Athlete 1,0000  1,0000  1,0000  36      0       0
     Automobile 1,0000  1,0000  1,0000  3       0       0
          Coach 1,0000  1,0000  1,0000  3       0       0
 Infrastructure 1,0000  1,0000  1,0000  22      0       0
 PoliticalParty 0,9848  0,9606  0,9726  195     3       8
     Politician 1,0000  0,8909  0,9423  49      0       6
PublicTransitSystem     0,9800  0,9423  0,9608  49      1       3
           Ship 1,0000  1,0000  1,0000  5       0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,9639  0,9639  0,9639  80      3       3
    SportsEvent 1,0000  0,9767  0,9882  42      0       1
   SportsLeague 0,9403  0,9545  0,9474  63      4       3
  SportsManager 1,0000  0,8333  0,9091  5       0       1
     SportsTeam 1,0000  1,0000  1,0000  33      0       0
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9825  0,9610  0,9716  616     11      25

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.22: All 3 Domains Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 100 Links

```
CRFClassifier tagged 4504 words in 1 documents at 2969,02 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       15
 PoliticalParty 0,9545  0,8077  0,8750  21      1       5
     Politician 1,0000  0,1111  0,2000  4       0       32
         Totals 0,9615  0,3247  0,4854  25      1       52

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\PoliticsFineGrainedTop10Links.tsv
```

Figure 3.23: All 3 Domains Fine Grained Top 300 Links Runned With Politics Fine Grained Top 10 Links

```
CRFClassifier tagged 40673 words in 1 documents at 4019,47 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       164
 PoliticalParty 0,9848  0,9606  0,9726  195     3       8
     Politician 1,0000  0,1536  0,2663  49      0       270
         Totals 0,9879  0,3557  0,5230  244     3       442

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\PoliticsFineGrainedTop100Links.tsv
```

Figure 3.24: All 3 Domains Fine Grained Top 300 Links Runned With Politics Fine Grained Top 100 Links

```
CRFClassifier tagged 2854 words in 1 documents at 2994,75 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  1,0000  1,0000  1       0       0
          Coach 1,0000  1,0000  1,0000  1       0       0
      SportsClub 1,0000  1,0000  1,0000  7       0       0
     SportsEvent 1,0000  1,0000  1,0000  1       0       0
    SportsLeague 1,0000  0,7500  0,8571  3       0       1
      SportsTeam 1,0000  1,0000  1,0000  1       0       0
          Totals 1,0000  0,9333  0,9655  14      0       1

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\SportFineGrainedTop10Links.tsv
```

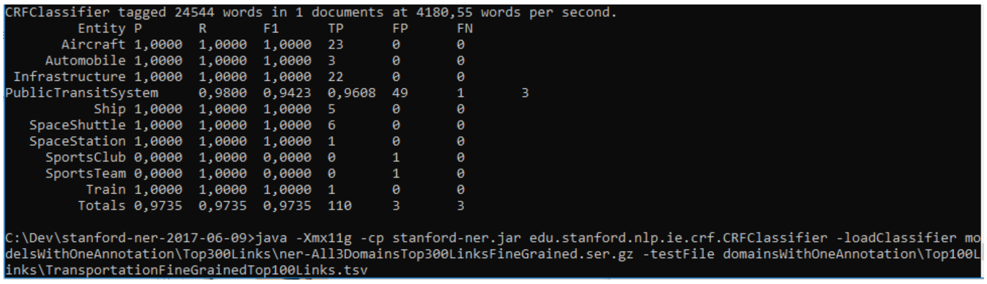
Figure 3.25: All 3 Domains Fine Grained Top 300 Links Runned With Sport Fine Grained Top 10 Links

```
CRFClassifier tagged 31872 words in 1 documents at 3691,88 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  1,0000  1,0000  36      0       0
          Coach 1,0000  1,0000  1,0000  3       0       0
      SportsClub 0,9634  0,9634  0,9634  79      3       3
     SportsEvent 1,0000  0,9767  0,9882  42      0       1
    SportsLeague 0,9403  0,9545  0,9474  63      4       3
   SportsManager 1,0000  0,8333  0,9091  5       0       1
      SportsTeam 1,0000  1,0000  1,0000  32      0       0
          Totals 0,9738  0,9701  0,9720  260     7       8

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\SportFineGrainedTop100Links.tsv
```

Figure 3.26: All 3 Domains Fine Grained Top 300 Links Runned With Sport Fine Grained Top 100 Links

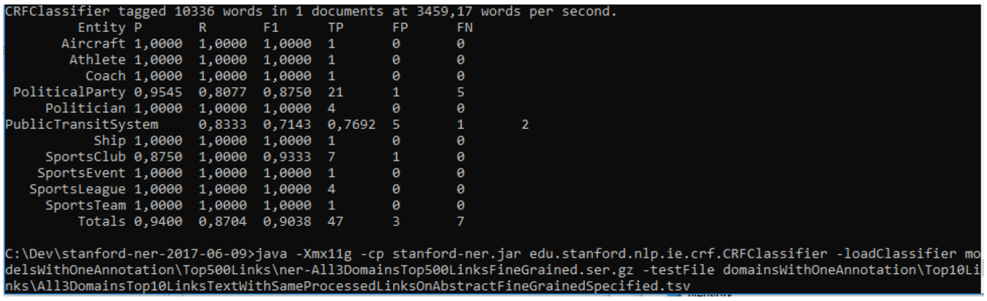
```
CRFClassifier tagged 2978 words in 1 documents at 2863,46 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  1       0       0
PublicTransitSystem     1,0000  0,8571  0,9231  6       0       1
           Ship 1,0000  1,0000  1,0000  1       0       0
         Totals 1,0000  0,8889  0,9412  8       0       1

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\TransportationFineGrainedTop10Links.tsv
```

Figure 3.27: All 3 Domains Fine Grained Top 300 Links Runned With Transportation Fine Grained Top 10 Links



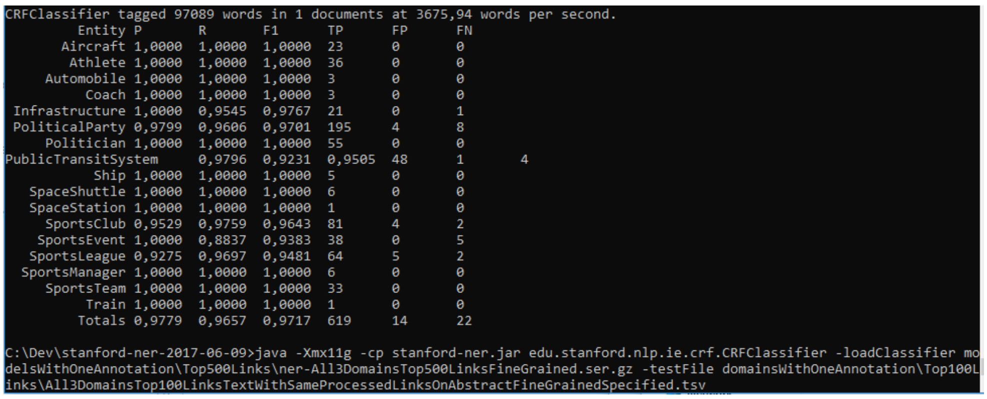
```
CRFClassifier tagged 24544 words in 1 documents at 4180,55 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  23      0       0
     Automobile 1,0000  1,0000  1,0000  3       0       0
 Infrastructure 1,0000  1,0000  1,0000  22      0       0
PublicTransitSystem     0,9800  0,9423  0,9608  49      1       3
           Ship 1,0000  1,0000  1,0000  5       0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,0000  1,0000  0,0000  0       1       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9735  0,9735  0,9735  110     3       3

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-All3DomainsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\TransportationFineGrainedTop100Links.tsv
```

Figure 3.28: All 3 Domains Fine Grained Top 300 Links Runned With Transportation Fine Grained Top 100 Links



```
CRFClassifier tagged 10336 words in 1 documents at 3459,17 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  1       0       0
        Athlete 1,0000  1,0000  1,0000  1       0       0
          Coach 1,0000  1,0000  1,0000  1       0       0
 PoliticalParty 0,9545  0,8077  0,8750  21      1       5
     Politician 1,0000  1,0000  1,0000  4       0       0
PublicTransitSystem     0,8333  0,7143  0,7692  5       1       2
           Ship 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,8750  1,0000  0,9333  7       1       0
    SportsEvent 1,0000  1,0000  1,0000  1       0       0
   SportsLeague 1,0000  1,0000  1,0000  4       0       0
     SportsTeam 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9400  0,8704  0,9038  47      3       7

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\All3DomainsTop10LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.29: All 3 Domains Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 10 Links



```
CRFClassifier tagged 97089 words in 1 documents at 3675,94 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  23      0       0
        Athlete 1,0000  1,0000  1,0000  36      0       0
     Automobile 1,0000  1,0000  1,0000  3       0       0
          Coach 1,0000  1,0000  1,0000  3       0       0
 Infrastructure 1,0000  0,9545  0,9767  21      0       1
 PoliticalParty 0,9799  0,9606  0,9701  195     4       8
     Politician 1,0000  1,0000  1,0000  55      0       0
PublicTransitSystem     0,9796  0,9231  0,9505  48      1       4
           Ship 1,0000  1,0000  1,0000  5       0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,9529  0,9759  0,9643  81      4       2
    SportsEvent 1,0000  0,8837  0,9383  38      0       5
   SportsLeague 0,9275  0,9697  0,9481  64      5       2
  SportsManager 1,0000  1,0000  1,0000  6       0       0
     SportsTeam 1,0000  1,0000  1,0000  33      0       0
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9779  0,9657  0,9717  619     14      22

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.30: All 3 Domains Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 100 Links

```
CRFClassifier tagged 270361 words in 1 documents at 2605,86 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  102     0       0
        Athlete 1,0000  0,9802  0,9900  198     0       4
     Automobile 1,0000  1,0000  1,0000  20      0       0
          Coach 1,0000  1,0000  1,0000  4       0       0
 Infrastructure 1,0000  0,9640  0,9817  107     0       4
 PoliticalParty 0,9718  0,9432  0,9573  482     14      29
     Politician 1,0000  1,0000  1,0000  139     0       0
PublicTransitSystem     0,9918  0,9718  0,9817  241     2       7
           Ship 1,0000  1,0000  1,0000  15      0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,9709  0,9625  0,9667  334     10      13
    SportsEvent 1,0000  0,8333  0,9091  55      0       11
   SportsLeague 0,9551  0,9707  0,9628  298     14      9
  SportsManager 1,0000  0,9615  0,9804  50      0       2
     SportsTeam 1,0000  0,9805  0,9902  151     0       3
          Train 1,0000  1,0000  1,0000  6       0       0
         Totals 0,9822  0,9642  0,9731  2209    40      82

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300L
inks\All3DomainsTop300LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.31: All 3 Domains Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 300 Links

```
CRFClassifier tagged 4504 words in 1 documents at 2429,34 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       15
 PoliticalParty 0,9545  0,8077  0,8750  21      1       5
     Politician 1,0000  0,1111  0,2000  4       0       32
         Totals 0,9615  0,3247  0,4854  25      1       52

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\PoliticsFineGrainedTop10Links.tsv
```

Figure 3.32: All 3 Domains Fine Grained Top 500 Links Runned With Politics Fine Grained Top 10 Links

```
CRFClassifier tagged 40673 words in 1 documents at 4212,64 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       164
 PoliticalParty 0,9799  0,9606  0,9701  195     4       8
     Politician 1,0000  0,1724  0,2941  55      0       264
         Totals 0,9843  0,3644  0,5319  250     4       436

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\PoliticsFineGrainedTop100Links.tsv
```

Figure 3.33: All 3 Domains Fine Grained Top 500 Links Runned With Politics Fine Grained Top 100 Links

```
CRFClassifier tagged 107491 words in 1 documents at 2506,26 words per second.
         Entity P       R       F1      TP      FP      FN
       Election 0,0000  0,0000  0,0000  0       0       322
 PoliticalParty 0,9718  0,9432  0,9573  482     14      29
     Politician 1,0000  0,1980  0,3305  136     0       551
PublicTransitSystem     0,0000  1,0000  0,0000  0       2       0
           Ship 0,0000  1,0000  0,0000  0       1       0
   SportsLeague 0,0000  1,0000  0,0000  0       1       0
         Totals 0,9717  0,4066  0,5733  618     18      902

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300L
inks\PoliticsFineGrainedTop300Links.tsv
```

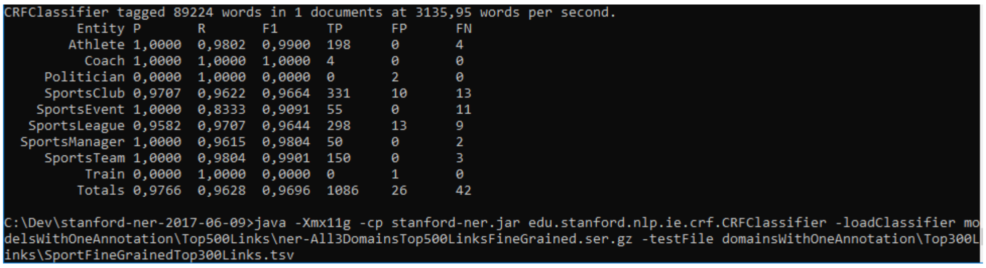
Figure 3.34: All 3 Domains Fine Grained Top 500 Links Runned With Politics Fine Grained Top 300 Links

```
CRFClassifier tagged 2854 words in 1 documents at 2679,81 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  1,0000  1,0000  1       0       0
          Coach 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,8750  1,0000  0,9333  7       1       0
    SportsEvent 1,0000  1,0000  1,0000  1       0       0
   SportsLeague 1,0000  1,0000  1,0000  4       0       0
     SportsTeam 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9375  1,0000  0,9677  15      1       0

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\SportFineGrainedTop10Links.tsv
```

Figure 3.35: All 3 Domains Fine Grained Top 500 Links Runned With Sport Fine Grained Top 10 Links

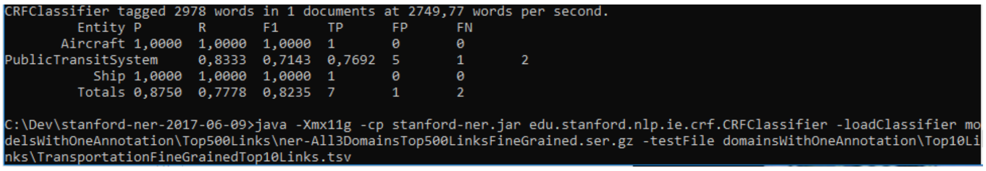
```
CRFClassifier tagged 31872 words in 1 documents at 4022,72 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  1,0000  1,0000  36      0       0
          Coach 1,0000  1,0000  1,0000  3       0       0
     SportsClub 0,9524  0,9756  0,9639  80      4       2
    SportsEvent 1,0000  0,8837  0,9383  38      0       5
   SportsLeague 0,9275  0,9697  0,9481  64      5       2
  SportsManager 1,0000  1,0000  1,0000  6       0       0
     SportsTeam 1,0000  1,0000  1,0000  32      0       0
         Totals 0,9664  0,9664  0,9664  259     9       9

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\SportFineGrainedTop100Links.tsv
```

Figure 3.36: All 3 Domains Fine Grained Top 500 Links Runned With Sport Fine Grained Top 100 Links



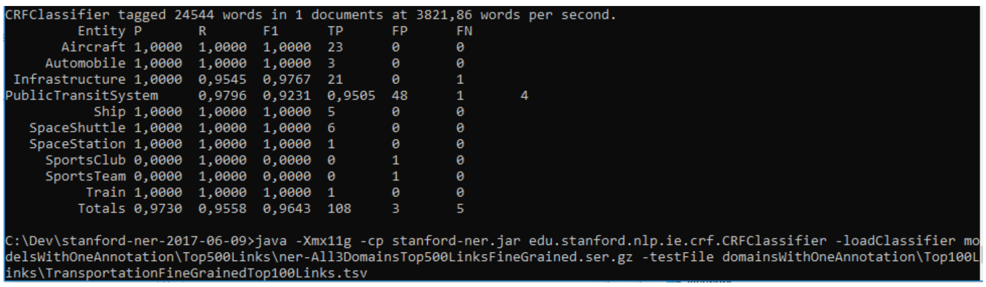
```
CRFClassifier tagged 89224 words in 1 documents at 3135,95 words per second.
         Entity P       R       F1      TP      FP      FN
        Athlete 1,0000  0,9802  0,9900  198     0       4
          Coach 1,0000  1,0000  1,0000  4       0       0
     Politician 0,0000  1,0000  0,0000  0       2       0
     SportsClub 0,9707  0,9622  0,9664  331     10      13
    SportsEvent 1,0000  0,8333  0,9091  55      0       11
   SportsLeague 0,9582  0,9707  0,9644  298     13      9
  SportsManager 1,0000  0,9615  0,9804  50      0       2
     SportsTeam 1,0000  0,9804  0,9901  150     0       3
          Train 0,0000  1,0000  0,0000  0       1       0
         Totals 0,9766  0,9628  0,9696  1086    26      42

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300L
inks\SportFineGrainedTop300Links.tsv
```

Figure 3.37: All 3 Domains Fine Grained Top 500 Links Runned With Sport Fine Grained Top 300 Links



```
CRFClassifier tagged 2978 words in 1 documents at 2749,77 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  1       0       0
PublicTransitSystem     0,8333  0,7143  0,7692  5       1       2
           Ship 1,0000  1,0000  1,0000  1       0       0
         Totals 0,8750  0,7778  0,8235  7       1       2

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top10Li
nks\TransportationFineGrainedTop10Links.tsv
```

Figure 3.38: All 3 Domains Fine Grained Top 500 Links Runned With Transportation Fine Grained Top 10 Links



```
CRFClassifier tagged 24544 words in 1 documents at 3821,86 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  23      0       0
     Automobile 1,0000  1,0000  1,0000  3       0       0
 Infrastructure 1,0000  0,9545  0,9767  21      0       1
PublicTransitSystem     0,9796  0,9231  0,9505  48      1       4
           Ship 1,0000  1,0000  1,0000  5       0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,0000  1,0000  0,0000  0       1       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9730  0,9558  0,9643  108     3       5

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100L
inks\TransportationFineGrainedTop100Links.tsv
```

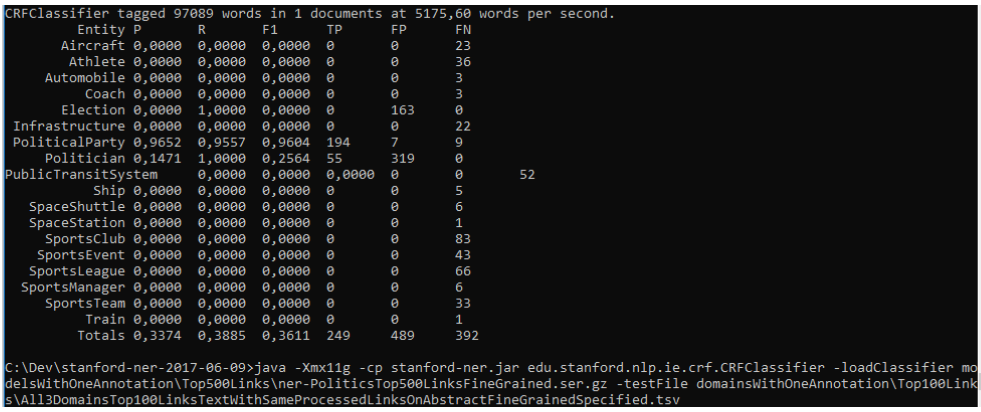
Figure 3.39: All 3 Domains Fine Grained Top 500 Links Runned With Transportation Fine Grained Top 100 Links

```
CRFClassifier tagged 73646 words in 1 documents at 3209,68 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  1,0000  1,0000  102     0       0
     Automobile 1,0000  1,0000  1,0000  20      0       0
 Infrastructure 1,0000  0,9640  0,9817  107     0       4
     Politician 0,0000  1,0000  0,0000  0       1       0
PublicTransitSystem     0,9917  0,9715  0,9815  239     2       7
           Ship 1,0000  1,0000  1,0000  14      0       0
   SpaceShuttle 1,0000  1,0000  1,0000  6       0       0
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,0000  1,0000  0,0000  0       3       0
     SportsTeam 0,0000  1,0000  0,0000  0       1       0
          Train 1,0000  1,0000  1,0000  5       0       0
         Totals 0,9860  0,9782  0,9821  494     7       11

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-All3DomainsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300L
inks\TransportationFineGrainedTop300Links.tsv
```

Figure 3.40: All 3 Domains Fine Grained Top 500 Links Runned With Transportation Fine Grained Top 300 Links

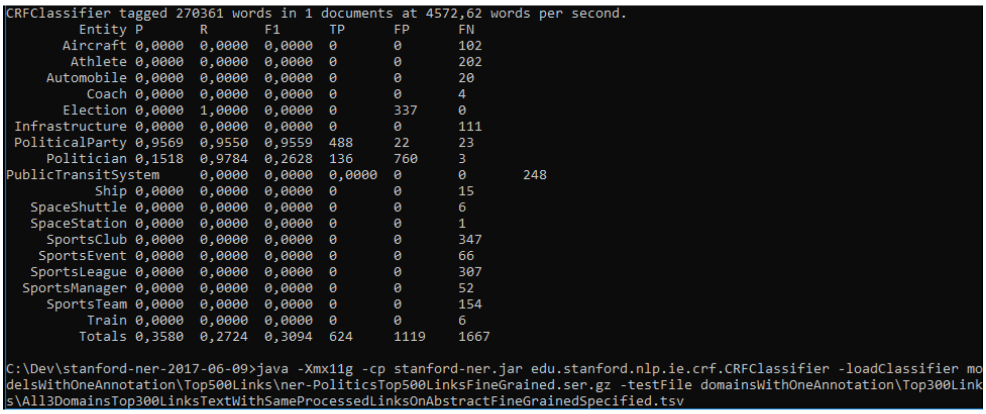
```
CRFClassifier tagged 97089 words in 1 documents at 7883,80 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       23
        Athlete 0,0000  0,0000  0,0000  0       0       36
     Automobile 0,0000  0,0000  0,0000  0       0       3
          Coach 0,0000  0,0000  0,0000  0       0       3
       Election 0,0000  1,0000  0,0000  0       136     0
 Infrastructure 0,0000  0,0000  0,0000  0       0       22
 PoliticalParty 0,7831  0,7291  0,7551  148     41      55
     Politician 0,1541  0,9273  0,2642  51      280     4
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       52
           Ship 0,0000  0,0000  0,0000  0       0       5
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  0,0000  0,0000  0       0       83
    SportsEvent 0,0000  0,0000  0,0000  0       0       43
   SportsLeague 0,0000  0,0000  0,0000  0       0       66
  SportsManager 0,0000  0,0000  0,0000  0       0       6
     SportsTeam 0,0000  0,0000  0,0000  0       0       33
          Train 0,0000  0,0000  0,0000  0       0       1
         Totals 0,3034  0,3105  0,3069  199     457     442

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-PoliticsTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100Link
s\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.41: Politics Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 100 Links



```
CRFClassifier tagged 97089 words in 1 documents at 5175,60 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       23
        Athlete 0,0000  0,0000  0,0000  0       0       36
     Automobile 0,0000  0,0000  0,0000  0       0       3
          Coach 0,0000  0,0000  0,0000  0       0       3
       Election 0,0000  1,0000  0,0000  0       163     0
 Infrastructure 0,0000  0,0000  0,0000  0       0       22
 PoliticalParty 0,9652  0,9557  0,9604  194     7       9
     Politician 0,1471  1,0000  0,2564  55      319     0
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       52
           Ship 0,0000  0,0000  0,0000  0       0       5
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  0,0000  0,0000  0       0       83
    SportsEvent 0,0000  0,0000  0,0000  0       0       43
   SportsLeague 0,0000  0,0000  0,0000  0       0       66
  SportsManager 0,0000  0,0000  0,0000  0       0       6
     SportsTeam 0,0000  0,0000  0,0000  0       0       33
          Train 0,0000  0,0000  0,0000  0       0       1
         Totals 0,3374  0,3885  0,3611  249     489     392

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-PoliticsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100Link
s\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.42: Politics Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 100 Links



```
CRFClassifier tagged 270361 words in 1 documents at 4572,62 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       102
        Athlete 0,0000  0,0000  0,0000  0       0       202
     Automobile 0,0000  0,0000  0,0000  0       0       20
          Coach 0,0000  0,0000  0,0000  0       0       4
       Election 0,0000  1,0000  0,0000  0       337     0
 Infrastructure 0,0000  0,0000  0,0000  0       0       111
 PoliticalParty 0,9569  0,9550  0,9559  488     22      23
     Politician 0,1518  0,9784  0,2628  136     760     3
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       248
           Ship 0,0000  0,0000  0,0000  0       0       15
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  0,0000  0,0000  0       0       347
    SportsEvent 0,0000  0,0000  0,0000  0       0       66
   SportsLeague 0,0000  0,0000  0,0000  0       0       307
  SportsManager 0,0000  0,0000  0,0000  0       0       52
     SportsTeam 0,0000  0,0000  0,0000  0       0       154
          Train 0,0000  0,0000  0,0000  0       0       6
         Totals 0,3580  0,2724  0,3094  624     1119    1667

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-PoliticsTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300Link
s\All3DomainsTop300LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

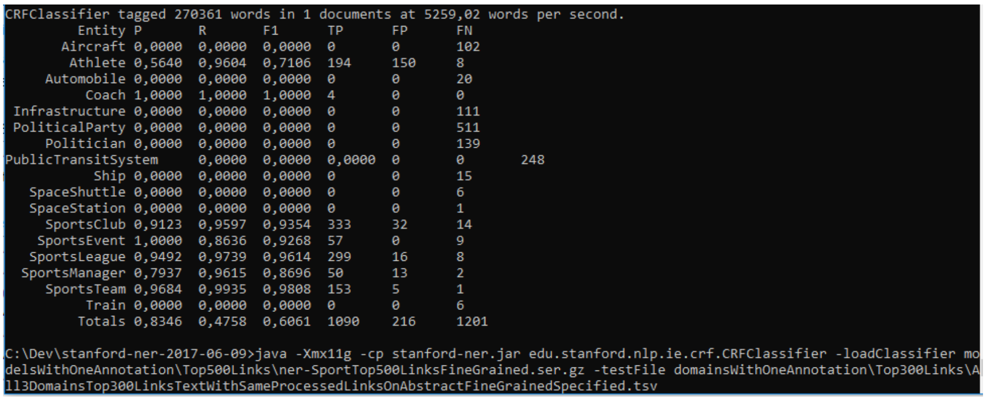
Figure 3.43: Politics Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 300 Links

```
CRFClassifier tagged 97089 words in 1 documents at 5454,44 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       23
        Athlete 0,2817  0,5556  0,3738  20      51      16
     Automobile 0,0000  0,0000  0,0000  0       0       3
          Coach 1,0000  0,6667  0,8000  2       0       1
 Infrastructure 0,0000  0,0000  0,0000  0       0       22
 PoliticalParty 0,0000  0,0000  0,0000  0       0       203
     Politician 0,0000  0,0000  0,0000  0       0       55
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       52
           Ship 0,0000  0,0000  0,0000  0       0       5
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,8795  0,8795  0,8795  73      10      10
    SportsEvent 0,9706  0,7674  0,8571  33      1       10
   SportsLeague 0,8852  0,8182  0,8504  54      7       12
  SportsManager 0,5000  1,0000  0,6667  6       6       0
     SportsTeam 0,9583  0,6970  0,8070  23      1       10
          Train 0,0000  0,0000  0,0000  0       0       1
         Totals 0,7352  0,3292  0,4547  211     76      430

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-SportTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100Links\A
ll3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.44: Sport Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 100 Links

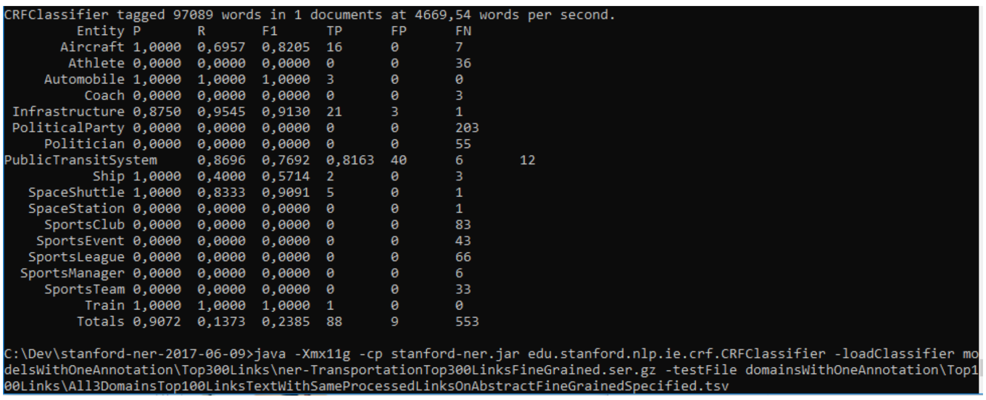
```
CRFClassifier tagged 97089 words in 1 documents at 5434,29 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       23
        Athlete 0,4359  0,9444  0,5965  34      44      2
     Automobile 0,0000  0,0000  0,0000  0       0       3
          Coach 1,0000  1,0000  1,0000  3       0       0
 Infrastructure 0,0000  0,0000  0,0000  0       0       22
 PoliticalParty 0,0000  0,0000  0,0000  0       0       203
     Politician 0,0000  0,0000  0,0000  0       0       55
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       52
           Ship 0,0000  0,0000  0,0000  0       0       5
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,8889  0,9639  0,9249  80      10      3
    SportsEvent 1,0000  0,8837  0,9383  38      0       5
   SportsLeague 0,8889  0,9697  0,9275  64      8       2
  SportsManager 0,4286  1,0000  0,6000  6       8       0
     SportsTeam 0,9697  0,9697  0,9697  32      1       1
          Train 0,0000  0,0000  0,0000  0       0       1
         Totals 0,7835  0,4009  0,5304  257     71      384

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-SportTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top100Links\A
ll3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.45: Sport Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 100 Links



```
CRFClassifier tagged 270361 words in 1 documents at 5259,02 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,0000  0,0000  0,0000  0       0       102
        Athlete 0,5640  0,9604  0,7106  194     150     8
     Automobile 0,0000  0,0000  0,0000  0       0       20
          Coach 1,0000  1,0000  1,0000  4       0       0
 Infrastructure 0,0000  0,0000  0,0000  0       0       111
 PoliticalParty 0,0000  0,0000  0,0000  0       0       511
     Politician 0,0000  0,0000  0,0000  0       0       139
PublicTransitSystem     0,0000  0,0000  0,0000  0       0       248
           Ship 0,0000  0,0000  0,0000  0       0       15
   SpaceShuttle 0,0000  0,0000  0,0000  0       0       6
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,9123  0,9597  0,9354  333     32      14
    SportsEvent 1,0000  0,8636  0,9268  57      0       9
   SportsLeague 0,9492  0,9739  0,9614  299     16      8
  SportsManager 0,7937  0,9615  0,8696  50      13      2
     SportsTeam 0,9684  0,9935  0,9808  153     5       1
          Train 0,0000  0,0000  0,0000  0       0       6
         Totals 0,8346  0,4758  0,6061  1090    216     1201

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-SportTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top300Links\A
ll3DomainsTop300LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.46: Sport Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 300 Links



```
CRFClassifier tagged 97089 words in 1 documents at 4669,54 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  0,6957  0,8205  16      0       7
        Athlete 0,0000  0,0000  0,0000  0       0       36
     Automobile 1,0000  1,0000  1,0000  3       0       0
          Coach 0,0000  0,0000  0,0000  0       0       3
 Infrastructure 0,8750  0,9545  0,9130  21      3       1
 PoliticalParty 0,0000  0,0000  0,0000  0       0       203
     Politician 0,0000  0,0000  0,0000  0       0       55
PublicTransitSystem     0,8696  0,7692  0,8163  40      6       12
           Ship 1,0000  0,4000  0,5714  2       0       3
   SpaceShuttle 1,0000  0,8333  0,9091  5       0       1
   SpaceStation 0,0000  0,0000  0,0000  0       0       1
     SportsClub 0,0000  0,0000  0,0000  0       0       83
    SportsEvent 0,0000  0,0000  0,0000  0       0       43
   SportsLeague 0,0000  0,0000  0,0000  0       0       66
  SportsManager 0,0000  0,0000  0,0000  0       0       6
     SportsTeam 0,0000  0,0000  0,0000  0       0       33
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9072  0,1373  0,2385  88      9       553

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top300Links\ner-TransportationTop300LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top1
00Links\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.47: Transportation Fine Grained Top 300 Links Runned With All 3 Domains Fine Grained Top 100 Links

```
CRFClassifier tagged 97089 words in 1 documents at 5359,00 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 1,0000  0,9565  0,9778  22      0       1
        Athlete 0,0000  0,0000  0,0000  0       0       36
     Automobile 1,0000  1,0000  1,0000  3       0       0
          Coach 0,0000  0,0000  0,0000  0       0       3
 Infrastructure 1,0000  1,0000  1,0000  22      0       0
 PoliticalParty 0,0000  0,0000  0,0000  0       0       203
     Politician 0,0000  0,0000  0,0000  0       0       55
PublicTransitSystem     0,9412  0,9231  0,9320  48      3       4
           Ship 1,0000  1,0000  1,0000  5       0       0
    SpaceShuttle 1,0000 0,6667  0,8000  4       0       2
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,0000  0,0000  0,0000  0       0       83
    SportsEvent 0,0000  0,0000  0,0000  0       0       43
   SportsLeague 0,0000  0,0000  0,0000  0       0       66
  SportsManager 0,0000  0,0000  0,0000  0       0       6
     SportsTeam 0,0000  0,0000  0,0000  0       0       33
          Train 1,0000  1,0000  1,0000  1       0       0
         Totals 0,9725  0,1654  0,2827  106     3       535

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-TransportationTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top1
00Links\All3DomainsTop100LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.48: Transportation Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 100 Links

```
CRFClassifier tagged 270361 words in 1 documents at 3536,53 words per second.
         Entity P       R       F1      TP      FP      FN
       Aircraft 0,9900  0,9706  0,9802  99      1       3
        Athlete 0,0000  0,0000  0,0000  0       0       202
     Automobile 1,0000  0,8500  0,9189  17      0       3
          Coach 0,0000  0,0000  0,0000  0       0       4
 Infrastructure 1,0000  0,9820  0,9909  109     0       2
 PoliticalParty 0,0000  0,0000  0,0000  0       0       511
     Politician 0,0000  0,0000  0,0000  0       0       139
PublicTransitSystem     0,9795  0,9637  0,9715  239     5       9
           Ship 1,0000  0,9333  0,9655  14      0       1
   SpaceShuttle 1,0000  0,6667  0,8000  4       0       2
   SpaceStation 1,0000  1,0000  1,0000  1       0       0
     SportsClub 0,0000  0,0000  0,0000  0       0       347
    SportsEvent 0,0000  0,0000  0,0000  0       0       66
   SportsLeague 0,0000  0,0000  0,0000  0       0       307
  SportsManager 0,0000  0,0000  0,0000  0       0       52
     SportsTeam 0,0000  0,0000  0,0000  0       0       154
          Train 1,0000  0,8333  0,9091  5       0       1
         Totals 0,9879  0,2130  0,3504  488     6       1803

C:\Dev\stanford-ner-2017-06-09>java -Xmx11g -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier mo
delsWithOneAnnotation\Top500Links\ner-TransportationTop500LinksFineGrained.ser.gz -testFile domainsWithOneAnnotation\Top3
00Links\All3DomainsTop300LinksTextWithSameProcessedLinksOnAbstractFineGrainedSpecified.tsv
```

Figure 3.49: Transportation Fine Grained Top 500 Links Runned With All 3 Domains Fine Grained Top 300 Links

3.4 Summary of results

3.4.1 Graphs

Conclusion

Bibliography

[1] Named Entity Recognition. Named Entity Recognition NER. Available from: https://en.wikipedia.org/wiki/Named-entity_recognition

[2] Michal Konkol. *Named Entity Recognition*. Master's thesis, University of West Bohemia in Pilsen, https://www.kiv.zcu.cz/site/documents/verejne/vyzkum/publikace/technicke-zpravy/2012/tr-2012-04.pdf, 2012.

[3] Wikipedia. Information extraction IE. Available from: https://en.wikipedia.org/wiki/Information_extraction

[4] Charles Sutton and Andrew McCallum. An Introduction to Conditional Random Fields for Relational Learning. *Introduction to Statistical Relational Learning. Edited by Lise Getoor and Ben Taskar*, 2006. Available from: http://people.cs.umass.edu/~mccallum/papers/crf-tutorial.pdf

[5] Charles Sutton and Andrew McCallum. An Introduction to Conditional Random Fields for Relational Learning. *Introduction to Statistical Relational Learning. Edited by Lise Getoor and Ben Taskar*, 2006. Available from: http://people.cs.umass.edu/~mccallum/papers/crf-tutorial.pdf

[6] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, 2005: pp. 363–370. Available from: http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf

[7] Wikipedia. DBpedia Spotlight. Available from: https://en.wikipedia.org/wiki/DBpedia#DBpedia_Spotlight

[8] Wikipedia. spaCy. Available from: https://en.wikipedia.org/wiki/SpaCy

[9] Wikipedia. GATE. Available from: https://en.wikipedia.org/wiki/General_Architecture_for_Text_Engineering

[10] Wikipedia. Resource Description Framework RDF. Available from: https://en.wikipedia.org/wiki/Resource_Description_Framework

[11] Usmanov Radmir. *NCollection, Transformation, and Integration of Data from the Web Services Domain.* Master's thesis, Czech Technical University in Prague, Faculty of Information Technology, https://dspace.cvut.cz/bitstream/handle/10467/72987/F8-DP-2017-Usmanov-Radmir-thesis.pdf, 2017.

[12] W3C. Resource Description Framework RDF. Available from: https://www.w3.org/RDF/

[13] W3C. Natural Language Processing Interchange Format NIF. Available from: https://www.w3.org/2015/09/bpmlod-reports/nif-based-nlp-webservices/#natural-language-processing-interchange-format-nif

[14] DBpedia. DBpedia Core. Available from: https://wiki.dbpedia.org/dbpedia-wiki

[15] DBpedia. DBpedia NIF Dataset. Available from: http://wiki.dbpedia.org/dbpedia-nif-dataset

[16] DBpedia. DBpedia Ontology. Available from: http://wiki.dbpedia.org/services-resources/ontology

[17] Wikipedia. Apache Jena. Available from: https://en.wikipedia.org/wiki/Apache_Jena

[18] Vivek Kulkarni and Yashar Mehdad and Troy Chevalier. Domain Adaptation for Named Entity Recognition in Online Media with Word Embeddings. *CoRR*, volume abs/1612.00148, 2016, 1612.00148. Available from: http://arxiv.org/abs/1612.00148

[19] Javier D. FernÃąndez and Miguel A. MartÃŋnez-Prieto and Claudio GutiÃľrrez and Axel Polleres and Mario Arias. Binary RDF Representation for Publication and Exchange (HDT). *Web Semantics: Science, Services and Agents on the World Wide Web*, volume 19, 2013: pp. 22–41. Available from: http://www.websemanticsjournal.org/index.php/ps/article/view/328

[20] Wikipedia. PageRank. Available from: https://en.wikipedia.org/wiki/PageRank

[21] Wikipedia. F1 score. Available from: https://en.wikipedia.org/wiki/F1_score

Appendix A

Retrieved types

A.1 POLITICS types

Parliament, Election,PoliticalParty, GeopoliticalOrganisation, Politician, Ambassador, Chancellor, Congressman, Deputy, Governor, Lieutenant, Mayor, MemberOfParliament, Minister, President, PrimeMinister, Senator, VicePresident, VicePrimeMinister, PoliticianSpouse, PersonFunction, PoliticalFunction, Profession, TopicalConcept and PoliticalConcept.

A.2 SPORT types

Types: Sport, firstOlympicEvent, footedness, TeamSport, SportsClub, HockeyClub, RugbyClub, SoccerClub, chairmanTitle, clubsRecordGoalscorer, fansgroup, firstGame, ground, largestWin, managerTitle, worstDefeat and NationalSoccerClub are grouped at SportsClub type.

Types: SportsLeague, AmericanFootballLeague, AustralianFootballLeague, AutoRacingLeague, BaseballLeague, BasketballLeague, BowlingLeague, BoxingLeague, CanadianFootballLeague, CricketLeague, CurlingLeague, CyclingLeague, FieldHockeyLeague, FormulaOneRacing, GolfLeague, HandballLeague, IceHockeyLeague, InlineHockeyLeague, LacrosseLeague, MixedMartialArtsLeague, MotorcycleRacingLeague, PaintballLeague, PoloLeague, RadioControlledRacingLeague, RugbyLeague, SoccerLeague, SoftballLeague, SpeedwayLeague, TennisLeague, VideogamesLeague and VolleyballLeague are grouped at SportsLeague type.

Types: SportsTeam, AmericanFootballTeam, AustralianFootballTeam, BaseballTeam, BasketballTeam, CanadianFootballTeam, CricketTeam, CyclingTeam, FormulaOneTeam, HandballTeam, HockeyTeam and SpeedwayTeam are grouped at SportsTeam type.

Types: Athlete, ArcherPlayer, AthleticsPlayer, AustralianRulesFootballPlayer, BadmintonPlayer, BaseballPlayer, BasketballPlayer, Bodybuilder, Boxer, AmateurBoxer, BullFighter, Canoeist, ChessPlayer, Cricketer, Cyclist, DartsPlayer, Fencer, GaelicGamesPlayer, GolfPlayer, GridironFootballPlayer, American-FootballPlayer, CanadianFootballPlayer, Gymnast, HandballPlayer, HighDiver, HorseRider, Jockey, LacrossePlayer, MartialArtist, MotorsportRacer, Motor-cycleRider, MotocycleRacer, SpeedwayRider, RacingDriver, DTMRacer, For-mulaOneRacer, NascarDriver, RallyDriver, NationalCollegiateAthleticAsso-ciationAthlete, NetballPlayer, PokerPlayer, Rower, RugbyPlayer, Snooker-Player, SnookerChamp, SoccerPlayer, SquashPlayer, Surfer, Swimmer, TableTen-nisPlayer, TeamMember, TennisPlayer, VolleyballPlayer, BeachVolleyballPlayer , WaterPoloPlayer, WinterSportPlayer, Biathlete, BobsleighAthlete , Cross-CountrySkier, Curler, FigureSkater, IceHockeyPlayer, NordicCombined, Skater, Ski_jumper, Skier, SpeedSkater, Wrestler, SumoWrestler, Athletics and cur-rentWorldChampion are grouped at Athlete type.

Types: Coach, AmericanFootballCoach, CollegeCoach and VolleyballCoach are grouped at Coach type.

Types: OrganizationMember, SportsTeamMember are grouped at Orga-nizationMember type.

Types: SportsManager, SoccerManager are grouped at SportsManager type.

Types: SportsEvent, CyclingCompetition, FootballMatch, GrandPrix, In-ternationalFootballLeagueEvent, MixedMartialArtsEvent, NationalFootball-LeagueEvent, Olympics, OlympicEvent, Race, CyclingRace, HorseRace, Mo-torRace, Tournament, GolfTournament, SoccerTournament, TennisTourna-ment, WomensTennisAssociationTournament, WrestlingEvent, SportCompe-titionResult, OlympicResult, SnookerWorldRanking, SportsSeason, Motor-sportSeason, SportsTeamSeason, BaseballSeason, FootballLeagueSeason, Na-tionalFootballLeagueSeason, NCAATeamSeason, SoccerClubSeason, Soccer-LeagueSeason and MotorSportSeason are grouped at SportsEvent type.

A.3 TRANSPORTATION types

Types: Aircraft, aircraftType, aircraftUser, ceiling, dischargeAverage, en-ginePower, engineType, gun, powerType, wingArea, wingspan and MilitaryAir-craft are grouped at Aircraft type.

Types: Automobile, automobilePlatform, bodyStyle, enginePower, engine-Type, powerType, transmission and AutomobileEngine are grouped at Auto-mobile type.

Types: Locomotive, boiler, boilerPressure and cylinderCount are grouped at Locomotive type.

Types: MilitaryVehicle, Motorcycle and SpaceStation are not grouped, this are leaved as it is.

Types: On-SiteTransportation, ConveyorSystem, Escalator and Moving-Walkway are grouped at On-SiteTransportation type.

Types: Rocket, countryOrigin, finalFlight, lowerEarthOrbitPayload, maidenFlight, rocketFunction, rocketStages and RocketEngine are gruped at Rocket type.

Types: Ship, captureDate, homeport, layingDown, maidenVoyage, numberOfPassengers, shipCrew and shipLaunch are grouped at Ship type.

Types: SpaceShuttle, contractAward, Crews, firstFlight, lastFlight, missions, numberOfCrew, numberOfLaunches and satellitesDeployed are grouped at SpaceShuttle type.

Types: Spacecraft, cargoFuel, cargoGas, cargoWater and rocket are grouped at Spacecraft type

Types: Train, locomotive, wagon and TrainCarriage are grouped at Train type.

Types: Tram, PublicTransitSystem, Airline and BusCompany are gruped at PublicTransitSystem type.

Types: Infrastructure, Airport, Port, RestArea, RouteOfTransportation, Bridge, RailwayLine, RailwayTunnel, Road, RoadJunction, RoadTunnel, WaterwayTunnel, Station, MetroStation, RailwayStation, RouteStop and TramStation are grouped at Infrastructure type.

Appendix B

Stanford NER properties file

Here is the example of one of the properties file that we use for creating model with all used flags:

```
# location of the training file
trainFile =
# location where you would like to save
#(serialize) your
# classifier; adding .gz at the end
#automatically gzips the file,
# making it smaller, and faster to load
serializeTo =

# structure of your training file;
# this tells the classifier that
# the word is in column 0 and the
# correct answer is in column 1
map = word=0,answer=1

# This specifies the order of the CRF:
# order 1 means that features
# apply at most to a class pair of
# previous class and current class
# or current class and next class.
maxLeft=1

# these are the features we'd like to
# train with
# some are discussed below, the rest can
# be understood by looking
# at NERFeatureFactory
useClassFeature=true
```

```
useWord=true
# word character ngrams will be included
# up to length 6 as prefixes
# and suffixes only
useNGrams=true
noMidNGrams=true
maxNGramLeng=6
usePrev=true
useNext=true
useDisjunctive=true
useSequences=true
usePrevSequences=true
saveFeatureIndexToDisk=true
useObservedSequencesOnly=true
# the last 4 properties deal with word
# shape features
useTypeSeqs=true
useTypeSeqs2=true
useTypeySequences=true
wordShape=chris2useLC
```

APPENDIX C

Contents of CD