Relation Extraction with Deep Learning Methods

Poleksić, Andrija

University of Rijeka – Faculty of Informatics and Digital Technologies

Information and Communication Systems

Andrija Poleksić

Relation Extraction with Deep Learning Methods

Master thesis

Mentor: prof. dr. sc. Sanda Martinčić-Ipšić

Rijeka, July 2023.

Sveučilište u Rijeci – Fakultet informatike i digitalnih tehnologija

Informacijski i komunikacijski sustavi

Andrija Poleksić

Izlučivanje relacija korištenjem metoda dubokog učenja

Diplomski rad

Mentor: prof. dr. sc. Sanda Martinčić-Ipšić

Rijeka, srpanj 2023.

FAKULTET INFORMATIKE I DIGITALNIH TEHNOLOGIJA
SVEUČILIŠTE U RIJECI

Rijeka, 25.4.2023.

Zadatak za diplomski rad

Pristupnik: Andrija Poleksić

Naziv diplomskog rada: Izlučivanje relacija korištenjem metoda dubokog učenja

Naziv diplomskog rada na eng. jeziku: Relation Extraction with Deep Learning Methods

Sadržaj zadatka:

Razvoj informacijskih tehnologija obilježen je hiperprodukcijom nestrukturiranih i polustrukturiranih podataka, pa tako i podataka u tekstovima. Ekstrakcija informacija, kao jedan od zadataka iz domene obrade prirodnog jezika, pokušava automatizirati njihovu obradu s ciljem izvlačenja podataka iz nestrukturiranog u strukturirani oblik. Tako je iz tekstualnih podataka moguće je izlučiti podatke o odnosima (relacijama) entiteta (subjekta i objekta) te se takav postupak naziva ekstrakcija ili izlučivanje relacija.

Cilj diplomskog rada je objasniti korake ekstrakcije relacija iz tekstova te njihovu povezanost s grafovima znanja. U provedbi zadatka će se koristiti pretrenirani modeli duboke neuronske mreže BERT i SCIBERT koji će se doučiti za zadatak ekstrakcije relacija. Analizirat i koristit će se postojeći skupovi podataka za ekstrakciju relacija te će se pripremiti podaci za znanstvenu domenu u obliku prikladnom za zadatak izlučivanja relacija na razini rečenice. U diplomskome radu je potrebno razviti i objasniti postupak ekstrakcije na odabranim dubokim modelima te vrednovati rezultate standardnim metrikama te usporediti performanse.

Mentor:
Prof. dr. sc. Sanda Martinčić-Ipšić

Voditeljica za diplomske radove:
Prof. dr. sc. Ana Meštrović

Komentor:

Zadatak preuzet: 27.4.2023.

(potpis pristupnika)

Fakultet informatike i digitalnih tehnologija
Radmile Matejčić 2, 51000 Rijeka, Hrvatska
http://inf.uniri.hr ured@inf.uniri.hr
Tel: +385 (0)51584700 | Fax: +385 (0)51 584 749

Abstract

As a result of technological progress (hardware and software), the amount of data is increasing and we are working to include all information that could be of importance. The task from the field of NLP that tries to automate this process, information extraction (IE), or rather, its subtask, relation extraction (RE), is studied in combination with the state of the art "*Bidirectional Encoder Representations from Transformers*" (BERT). In this thesis, first the basic concepts of Deep Learning and Natural Language Processing (NLP) with a brief historical overview are introduced. Then, the summary on datasets suitable for the task of extracting sentence-level relations is presented. Based on the presented datasets, a new dataset for scientific RE, combo160, is created and used to fine-tune the BERT and SciBERT models for the task. From the results, it can be inferred that thematically similar (here: scientific) pretrainig corpora can indeed improve the performance of the later fine-tuned models for RE on scientific data.

Keywords: Relation extraction, relation classification, BERT, SciBERT, scientific dataset

Sažetak

Izlučivanje relacija korištenjem metoda dubokog učenja

U ovom diplomskom radu najprije se uvode osnovni koncepti dubokog učenja i računalne obrade prirodnog jezika s kratkim povijesnim pregledom. Zatim je prikazan sažetak skupova podataka prikladnih za zadatak izlučivanja relacija na razini rečenice. Na temelju predstavljenih skupova podataka kreiran je novi skup podataka za izlučivanje relacija znanstvene tematike, combo160, koji se koristi za doučavanje BERT i SciBERT modela za dani zadatak izlučivanja relacija.

Iz rezultata se može zaključiti da tematski slični (ovdje: znanstveni) korpusi korišteni za doučavanje pretreniranih BERT odnosno SciBERT modela doista poboljšavaju izlučivanje relacija na znanstvenim tekstovima.

Ključne riječi: izlučivanje relacija, klasifikacija relacija, BERT, SciBERT, skup znanstvenih podataka

Contents

1 Introduction

Natural language[1], as a constantly evolving medium for communication, transferring of ideas, thoughts and knowledge in human interactions, accounts for most of the data flowing through the Internet. As a result of technological (hardware and software) advancements, data flow volumes are increasing (i.e. big data phenomena evolved), and efforts to obtain all the relevant information are significant. Although we can understand natural (spoken) language effortlessly, it remains a difficult but necessary task to automate it with the help of computers.

Natural language processing (NLP), as a research area of computer science and artificial intelligence (AI), focuses on the design and analysis of computational algorithms and representations for processing natural human language [2]. NLP has numerous tasks and applications, including machine translation, text summarization, information retrieval, sentiment analysis, text classification, topic modeling, and **information extraction**. For a better overview, Figure 1 is presented below, followed by a brief historical review of NLP.

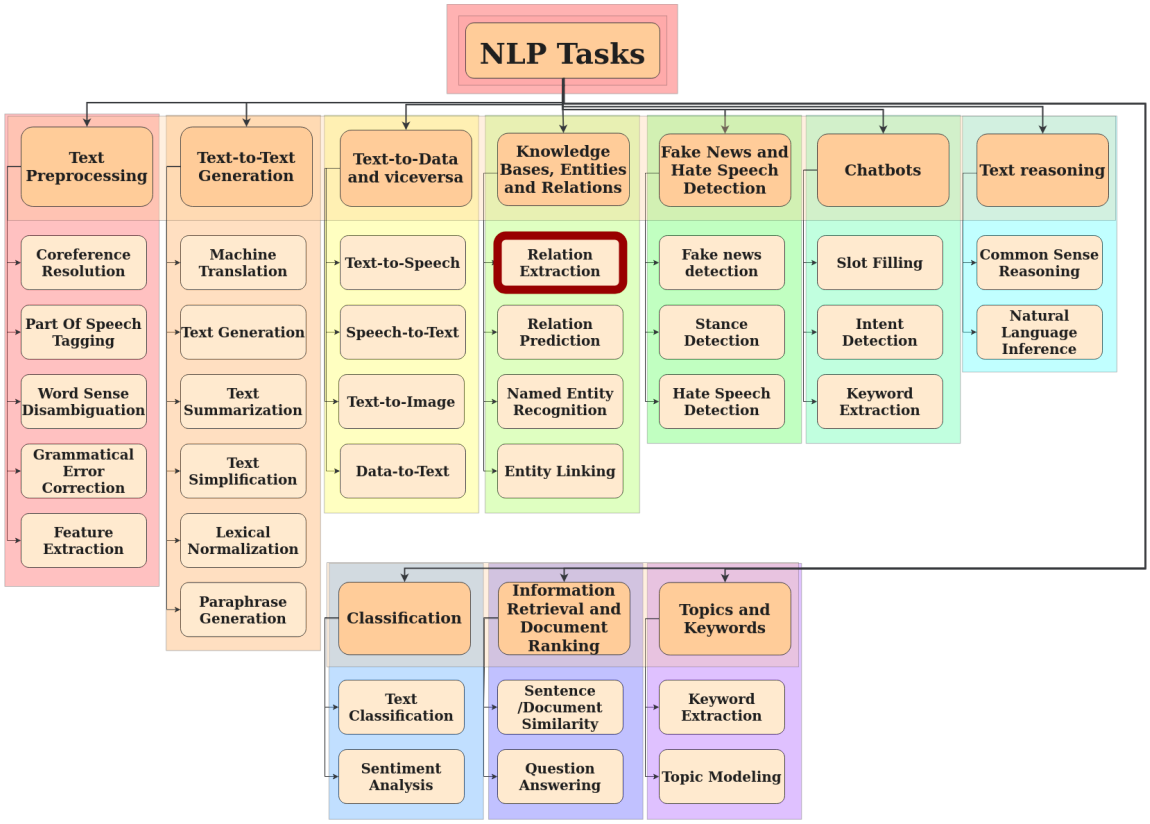

Figure 1: **NLP tasks**: Decomposition into main subareas-tasks, adapted from work of Fabio Chiusano: *https://medium.com/nlplanet/two-minutes-nlp-33-important-nlp-tasks-explained-31e2caad2b1b*

[1] "*A language that has developed and evolved naturally, through use by human beings, as opposed to an invented or constructed (artificial) language, as a computer programming language (often used attributively)*" [1].

It is important to note that some of these tasks, such as machine translation, speech recognition (speech-to-text), and conversational robots (chatbots), had their earliest prototypes in the 1950s. Inevitably, the prototypes were unsuccessful because the tasks were far more complex than originally anticipated, as evidenced by the 1954 Georgetown experiment [3], which was criticized by the Automatic Language Processing Advisory Committee (ALPAC) in 1966, leading to a significant decrease in government funding. To limit the scope of this work, earlier attempts at machine translation [4] and notable examples from other tasks, such as SHRDLU [5] (text reasoning) and ELIZA [6] (chatbot) are mentioned only.

The latter part of the 20th century was characterized by conceptual systems based on complex sets of handwritten rules and predefined **ontologies** for natural language processing. The tedious task of manual crafting of rules was initially automated thanks to technological (increase in computing capacity) and theoretical (Chomskian theories [7]) advances. During this time, NLP relied primarily on well-known and established machine learning algorithms such as decision trees. In the early 1990s, it became clear that the rigid rules generated by simple machine learning algorithms were not good enough. At the same time, with the availability of larger text corpora and advances in machine learning, a promising concept of **statistical NLP** emerged [8].

Machine learning procedures can make use of statistical inference algorithms to produce models that are more robust and reliable to unknown or erroneous inputs. Unlike language processing systems, which were designed by hand-coding a set of rules, such as grammars or devising heuristic rules for derivation, the machine learning paradigm instead requires the use of statistical inference to automatically learn such rules through the analysis of large numbers of typical real-world examples [9] [8]. For this reason, the latter can be made more accurate simply by providing more input data. The others, on the other hand, can only be made more accurate by increasing the complexity of the rules, which is a much more difficult task [10]. C. D. Manning and J. Hirschberg in [9] state that the central finding of new statistical approach to NLP is that simple methods using words, **part-of-speech (POS)** sequences, or simple templates often achieve notable results when trained on large quantities of data.

The majority of research has already shifted to statistical and probabilistic models as we enter the twenty-first century. Slow but consistent research progress was made between the years 2000 and 2014, including the use of neural networks (the first neural language models) and dense vector representations of words. Followed by the first implementations of recurrent neural networks (RNN) and later, a more sophisticated version, the long short-term memory (LSTM) neural networks used for sequence-to-sequence tasks such as machine translation. The research community quickly realized the potential of RNNs, which sparked a boom in RNN-based applications that quickly replaced earlier state-of-the-art technologies. A year later, in "*Neural Machine Translation by Jointly Learning to Align and Translate*" Bahdanau et al. introduced the attention mechanism [11]. The concept gained a lot of interest from the research community, even resulting in Google's change of course towards a new neural machine translation model [12] in 2016 with only 500 lines of code, replacing the previous 500,000 lines of phrase-based machine translation, as stated by Jeff Dean [13].

The famous scientific paper "*Attention Is All You Need (2017)*" by Vaswani et al. [14] introduced the new attention-only architecture called **Transformer**. Transformers

became the new state of the art and have replaced RNN-based approaches in recent years. Meanwhile, the concept of **pretrained language models** (PLM), first proposed by Dai and Le [15] in 2015, has received much attention. The basic idea is that by running the language model (LM), we can obtain contextualized word representations that can then be used as a base layer in a supervised neural network for any task. With this in mind, one of the first attempts at neural language modeling (NLM)[2] for the later fine-tuning on downstream tasks were the ELMo model of Peters et al. [16], which is based on the biLSTM architecture and ULMfit model of Howard and Ruder [17] based on transformer architecture. Later in the same year, the widely successful BERT model was introduced in paper "*BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*" of Devlin et al. [18]. To better understand the evolution and the nomenclature of the concept of large language models (LLM) which stem from the PLM idea, the reader is encouraged to inspect the work of Zhao et al. [19].

1.1 Motivation and contribution

In this work, the BERT model is used as a base model for fine-tuning on the relation extraction objective due to its proven good results on the majority of NLP tasks. In addition, a performance comparison of two models **SciBERT and BERT** a the created dataset called **combo160** is performed to provide insight into contributions of using different corpora during pretraining. Primarily, this work tests the hypothesis "Pretrained models should differ in their results when the pretraining set is thematically equivalent to the later fine-tuned objective."; or, to further specify and reframe it into the research question, "**Will the SciBERT model perform better than the BERT model on the relation extraction task on scientific corpora**". Although the final conclusion cannot be drawn based solely on this isolated experiment, it certainly reinforces the insights into the answer. Below, the major contributions of this thesis are listed:

- Creation (selection) of the new scientifically oriented dataset useful for further benchmarks on reletion extraction- **combo160**;
- Two models (BERT and SciBERT) fine-tuned for RE available for use through **OpenNRE toolkit** [20];

[2]Distinction is made between statistical language models (SLM) developed from the period of 1990 to 2010 (based on machine learning approaches and mainly referred to as n-gram models) and neural language models (NLM) based strictly on neural networks. The PLM concept could be observed as a subclass of NLM.

1.2 The structure of thesis

In the next section 2 reader is introduced to the task of relation extraction and its historical background. Together with the motivation, a connection with the construction of knowledge graphs is established. The following section 3 introduces the basics of neural networks before the detailed section 4 discusses the transformer architecture and the architectures of BERT and SciBERT used in this work. Next the summary of the datasets used is provided in the Section 5. Section 6 presents the newly created dataset **combo160** and the experimental setup. The final Sections of the paper, Sections 7 and 8, cover the results, and conclusion respectively.

2 Relation extraction

The sources of unstructured and semi-structured text on the Web flood the daily user, whether news journals, email communications, blogs, forums, online encyclopedias, government documents, or papers from the scientific community. We are struggling to find a way to automatically structure the information that lies among these infinite sources of data. **Information extraction (IE)** is the task (field) of NLP that deals with the problem of extracting information from unstructured texts, as can be seen from the definition from Sunita Sarawagi in [21]:

> "*Information Extraction refers to the automatic extraction of structured information such as entities, relationships between entities, and attributes describing entities from unstructured sources.*"

IE is not a trivial problem, it is divided into a number of smaller tasks, which include named entity recognition (NER), coreference resolution, and relation extraction. Let us now define the main topic of this thesis. **Relation extraction (RE)** is the subtask of information extraction that consists of identifying mentions of the relations of interest in each sentence, paragraph, or larger unit of text. It usually involves extracting the relation between two or more (named) entities [22]. **Named entities (NE)** are traditionally detected by first applying NER and then RE. The result can be defined as a word or phrase representing a specific real-world object. To further clarify, RE model takes unstructured text with/without marked entities as an input and outputs the triples that usually resemble:

(subject (named entity), **relation** (relation type), **object** (named entity)**)**

In the Table 1 below we can see several examples of extracted relations. Note that the underlined parts represent the entities detected by the NER model or jointly in the relation extraction model as of recent trends.

Sentence	Relation
Relation extraction (RE) is the subtask of information extraction.	subtask_of
There is a house way down in New Orleans	location_of
The town blossomed in the 18th and 19th centuries with the development of roads to the seaside and waterways along the Kupa River.	near_body_of_water
Though Kid A divided listeners, it was later named the best album of the decade by multiple outlets.	"named the best album of the decade by"
Now that the first person interface has become the design of choice for the industry, Id will need to find new innovations.	"has become"

Table 1: **Examples of relations in sentences**: First three rows depict RE with finite set of relations, while latter two represent RE without previously defined schema.

Upon closer inspection, we can see similarities in nomenclature between the first three examples, and the same is true for the other two examples at the bottom of the Table. The Table previews two different understandings of the relation extraction task: the first approach considers multi-class classification on the finite set of relations, and the second approach refers to the much harder task of identifying relations without a strictly defined template of what the relation should look like. The first technique is erratically specified also as the relation classification (RC). For further insight into the definition of relation extraction, the reader is encouraged to examine the research of Bassignana and Plank [23], where the definition of the RE task is revisited along with a comprehensive survey on RE datasets. In the next paragraph, a brief overview of the evolution of RE methods is provided.

Relation extraction, compared to other NLP tasks is a relatively novel discipline, with the first attempts considered dating back to the beginning of the 21st century. According to Hun et al. [24] the evolution of RE is roughly divided into 3 parts:

- **Pattern extraction models**

  The first models rely on sentence analysis tools that identify syntactic elements in the text, whereupon pattern rules are automatically generated. Pattern rules are error-prone and therefore require a high level of intervention by human experts.

- **Statistical relation extraction models**

  Along with the rest of NLP, RE has evolved to statistical models that significantly reduce the need for human intervention and provide better coverage of the task. The authors suggest some approaches that are used: feature-based methods, kernel-based methods, graphical methods, and embedding-model inspired methods.

- **Neural relation extraction models**

  With the increasing usability and popularity of **neural networks (NN)** (discussed in section 3), especially with the use of GPU [25], methods incorporating them quickly overtook the field of RE. The first phase of use included various NN architectures that attempted to capture the semantics of text, such as recursive neural networks, convolutional neural networks (CNN), attention-based neural networks, and recurrent neural networks (RNN), which dominated the field until the advent of the transformer architecture and approaches relying on pretrained (large) language models.

Relation extraction, as done by the majority of the research community, is usually tackled in two main setups:

- **Pipeline approach**

  In the pipeline approach, the NER and RE tasks are trained separately, therefore the RE model expects already extracted entities in the input text, which may be of lower quality, propagating the error. Although the pipeline approach suffers from error propagation, it is easy to implement and yields good results, as shown in the work of Nguyen et al. [26], Ale et al. [27], and Zhou et al. [28].

- **Joint entity and relation approach**

  In joint entity and relation extraction, the model is trained to perform both tasks simultaneously while benefiting from the use of interrelated signals. This approach has attracted a lot of attention in recent years and has provided new state-of-the-art results in various benchmarks. Some examples are the work of P. L. Huguet Cabot and R. Navigli [29], which define the RE task as seq2seq generation, and Tang et al. [30] which uses matrix-like interaction maps to effectively represent relations and NE together.

Before moving to the definition of neural networks and advanced architectures such as transformers, a discussion on evaluation and the primary motivation for RE is presented.

2.1 Relation extraction evaluation

Relation extraction and its associated named entity recognition task are relatively difficult to evaluate, especially when two tasks are trained in a joint scenario (sometimes referred to as end-to-end RE). Taillé et al. [31] discuss this problem further and address the common errors that occur even in state-of-the-art models for joint RE and NER tasks. In this thesis, the NER task is neglected, i.e., the model expects tagged inputs for further standard multiclass classification. For this task, seven metrics are presented: Micro and Macro versions of precision, recall and F1-score, and the standard accuracy metric. These metrics rely on a confusion matrix, a cross table that records the number of occurrences between actual classes and classes predicted by a model. In Figure 2 the example of two class classifications confusion matrix is presented, where:

- True positive (**TP**) - number of times a model correctly predicted the positive class
- True negative (**TN**) - number of times a model correctly predicted the negative class
- False positive (**FP**) - number of times a model falsely predicted the positive class
- False negative (**FN**) - number of times a model falsely predicted the negative class

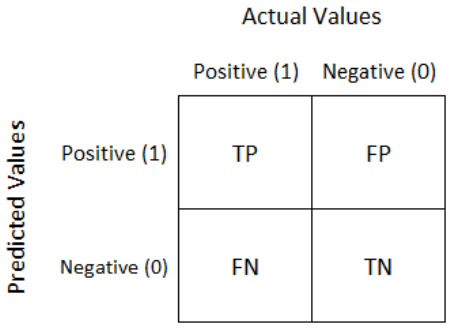

Figure 2: **Binary confusion matrix**: Cross table that records the number of occurrences between actual classes (values) and classes (values) predicted by a model, where TP and TN count values that are correctly predicted by model as positive and negative respectively. Adopted from: *https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62*

For brevity, the labels (TP, TN, FP, FN) are explained on binary classification task, a similar concept remains with the increasing number of classes with further elaboration on the calculation of each of the four explained values. An in-depth review of TP, TN, FP, and FN calculations on the multiclass classification work of Grandini et al. [32] is recommended.

2.1.1 Accuracy

Accuracy, the metric that rewards models with the most correct predictions (TP and TN) is computed as:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

This metric works desirably for balanced datasets, i.e. datasets that have a similar distribution of instances in all classes.

2.1.2 Precision

Precision is defined as the ratio of TP predictions over the sum of TP and FP predictions. Calculation on binary classification is straightforward:

$$precision = \frac{TP}{TP + FP}.$$

Precision can also be used on multiclass classification with two different approaches: for each class individually (tediously observing binary classification for each class) and with averaging across all classes (micro and macro averaging). Of two, the latter is used both

for precision and recall calculation, also extending to F1-score. There are two standards used for averaging along classes:

- **Macro precision**

  Macro precision is calculated similar to calculating precision for each class individually in a multiclass problem. To calculate precision in this manner, for each class, we treat all other classes as negative, leaving us with one positive class consisting of a single class, and one negative class consisting of the rest of the classes. After calculating precision for each class in this manner, per-class precision is averaged as:

$$precision_{MACRO} = \frac{precision_{C1} + precision_{C2} + ... + precision_{CN}}{N}$$

  where N is the number of classes denoted by C1 to CN.

- **Micro precision**

  Micro precision uses the same base concept. For each class, in the same manner, as the first step of Macro precision, we sum all TP, FP, and FN instances per class. After this process, we sum all of the TPs per class together into a Total True Positive sum, the same is done for FP and FN. Based on these three total sums, the precision is then calculated:

$$\begin{aligned} precision_{MICRO} &= \frac{TP_1 + TP_2 + ... + TP_N}{TP_1 + FP_1 + TP_2 + FP_2 + ... + TP_N + FP_N} \\ &= \frac{TP_{TOTAL}}{TP_{TOTAL} + FP_{TOTAL}}. \end{aligned}$$

2.1.3 Recall

The recall is defined as the fraction of TP elements divided by the total number of positively classified units, i.e. TP and FN elements. The recall is then defined as:

$$recall = \frac{TP}{TP + FN}.$$

When working with multiple classes, the procedure is similar to the precision:

- **Macro recall**

$$recall_{MACRO} = \frac{recall_{C1} + recall_{C2} + ... + recall_{CN}}{N}$$

where N is the number of classes denoted by C1 to CN.

- **Micro recall**

$$recall_{MICRO} = \frac{TP_1 + TP_2 + ... + TP_N}{TP_1 + FN_1 + TP_2 + FN_2 + ... + TP_N + FN_N}$$
$$= \frac{TP_{TOTAL}}{TP_{TOTAL} + FN_{TOTAL}}$$

2.1.4 F1-score

F1-score combines precision and recall of the model as their harmonic mean:

$$F1 = \frac{2(precision * recall)}{precision + recall}.$$

Since the metrics on which the F1-score rely have different approaches, mainly **micro** and **macro** averaging, the same classification is present for the F1-score. Thus, there exists a micro and macro version of the F1-score. F1-score is mainly used to compare models' performances while considering both, recall and precision.

2.2 Knowledge graphs and RE

Let us define knowledge as the understanding of, or a piece of information about a subject gathered through experience or study, known by one or more persons [33]. Knowledge can be gathered from various sources and in various forms, including large corpora of text (natural language). If we observe the sentence: "*One of Nazor's main prose works is the extensive novel Loda the Shepherd (Pastir Loda) (1938).*" we can safely deduce following facts:

- "Loda the Shepherd" is a novel.
- "Loda the Shepherd" was written in 1938.
- Nazor wrote "Loda the Shepherd".

We can say that we have gained knowledge about the subject of Nazor or the novel "Loda the Shepherd". Let us suppose that we want to store this knowledge. One of the possible options, in this case, would be the knowledge graph. **Knowledge graph (KG)**[3] is a graph-like structure[4] with entities (e.g., *Nazor, novel, or "Loda the Shepherd"*) connected via the edges representing the relations (e.g. *"Loda the Shepherd"* **is a** *novel.*). With this setup, it is clear that relation triples can be viewed as the single edge, together with the incident nodes.

Thus, knowledge graphs can be constructed automatically from extracted relations, and relations can be extracted based on the predefined set of edge types (relations) in a KG. Relations in a graph can have various properties, such as a unique direction of the relation or mathematical properties like symmetry or transitivity. Although we can say that Nazor wrote "Loda the Shepherd", the converse case cannot be established, since it is impossible for a novel to write a person. This is obvious to the reader, but it must be attended and clearly defined when it comes to the relations between the entities in KG. In addition, other relations that are not explicitly stated in the sentences can be automatically extracted from the graph using the knowledge graph completion (KGC) procedure (e.g., Novel is a book. → "Loda the Shepherd" is a book).

RE datasets ([36], [37], [38], [39], [24], [40]) often rely on the distant supervision technique rather than the time- and resource-consuming but less noisy, human annotation. Distant supervision makes use of an existing knowledge database (i.e., knowledge graph), such as Freebase [41] or later Wikidata [42], to collect examples of relations of interest. That further emphasizes the intertwine nature of KG and RE fields. Similarly, RE is often used to improve KGs, as can be seen in the work of Pingle et al. [43] where RE is used for the improvement of cybersecurity KG. Also, some researches, such as works of Yu et al. [44], Li et al. [45], and Luan et al. [46] preview the implementation of relation extraction for KG construction.

[3]The definition of the knowledge graph is simplified in this paper to emphasize the common features of KG and RE. For further exploration, the work of Hogan et al. [34] and Ji et al. [35] are suggested.

[4]In a restricted but very common sense of the term, a graph is an ordered pair G=(V,E) consisting of a set of vertices V (nodes) and a set of edges E (links, lines) that are unordered pairs of vertices (i.e. one edge is connected to two distinct vertices).

3 Neural networks

Neural networks consist of layers of artificial neurons. The inspiration for this design comes from the neurons in the human brain, as described in the 1957 technical report [47] by F. Rosenblatt. The Figure 3 shows the visual example of a simple neuron, a perceptron. A perceptron has one or more inputs labelled x_1, x_2, ... , x_n and the corresponding weights denoted by w_1, w_2, ... , w_n.

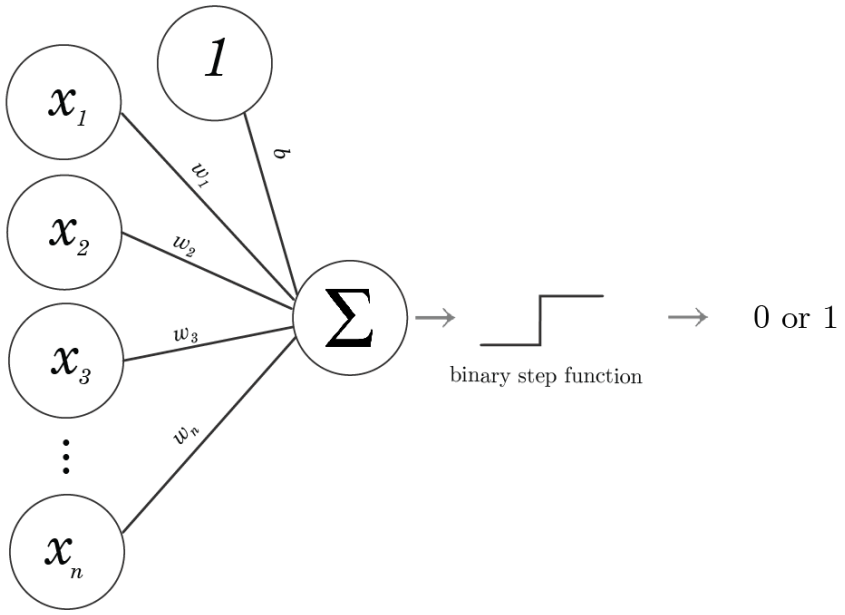

Figure 3: **Perceptron**: Visual example of a simple artificial neuron, the Perceptron, consisting of x_1, x_2, ... , x_n inputs, corresponding weights w_1, w_2, ... , w_n, and an activation function (here: binary step function). Adopted from: *https://ai.plainenglish.io/the-rise-and-fall-of-the-perceptron-c04ae53ea465*

Considering a set of inputs and the corresponding weights as vectors X and W, respectively, we can define the output of the neuron as follows:

$$f(X) = \begin{cases} 0, & W \cdot X + b \leq 0 \\ 1, & W \cdot X + b > 0 \end{cases},$$

where function $f(X)$ is called the activation function, and term b represents the threshold or bias of a neuron. This simple neuron can only classify the data linearly, but with a larger number of neurons stacked into layers and more complex and derivable activation functions, we can solve complex, non-linear tasks. This simple idea incorporates the core of neural networks. To limit the scope of this thesis, no further definitions of neural network architectures (except basic transformer and BERT) are provided.

3.1 Feedforward neural network

A feedforward network is a neural network with multiple layers, in its simplest form with an input layer, a hidden layer, and an output layer. As the name implies, each layer receives calculations (outputs) strictly from a previous layer, with the final output layer shaped for a specific task, such as binary classification. Figure 4 shows an example of a feedforward neural network, where each circle represents an artificial neuron similar to the neuron shown in Figure 3.

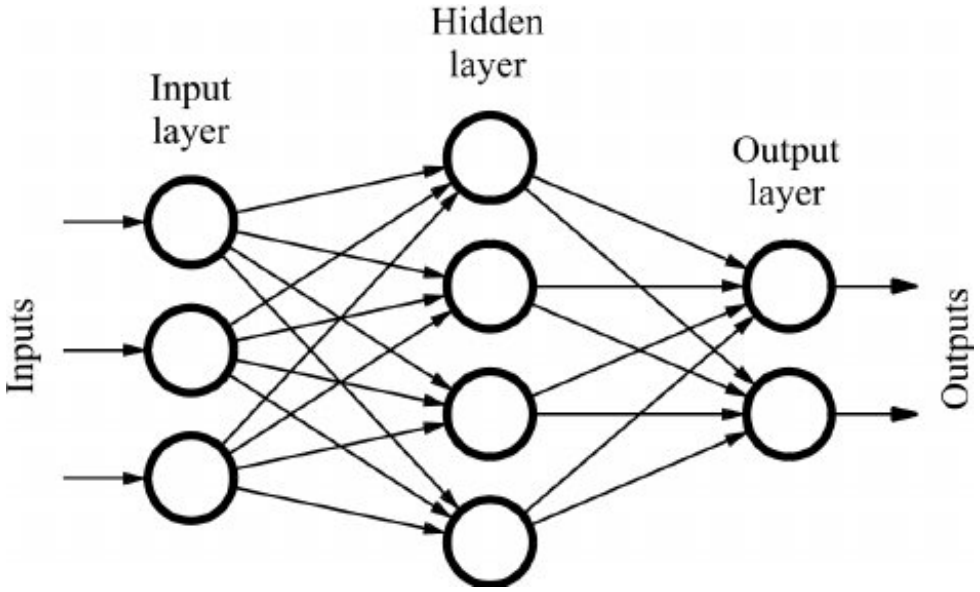

Figure 4: **Feedforward neural network**: A simple feedforward NN consisting of three layers, input, hidden, and output layer. Adopted from: *https://www.researchgate.net/publication/234055177_Computational_Methods_and_Optimization/figures?lo=1*

3.2 Residual connection

Residual connections or skip connections, a concept that gained great popularity after it was introduced in the work of He at al. [48], is a widely used approach when dealing with networks with a larger number of layers of artificial neurons (i.e., deep neural networks). Skip connection mitigates the troubling problems of vanishing and exploding gradients by connecting non-sequential layers of the network, creating multiple paths for the values to pass. As pointed out by Veit et al. [49], "*Residual networks avoid the vanishing gradient problem by introducing short paths which can carry gradient throughout the extent of very deep networks.*"

4 Transformers

As stated in the introduction (Section 1), work of Vaswani et al. [14] quickly overtook the NLP scene with the new architecture that is able to ingest and contextualize an unlimited sequence length, minding the prominence of the enabling hardware underneath. As opposed to previous state-of-the-art neural models for the majority of sequence-based NLP tasks, mainly memory-enriched RNN architectures, such as long short-term memory (LSTM) [50] and gated recurrent unit (GRU) [51] the transformer relies on attention mechanism only, mitigating the need for the recurrent connections. Recurrent connections, due to their sequential nature suffer from low parallelizability, limiting implementation at larger scales. Transformers, on the other hand, do not suffer from the same problem.

Transformers are made of transformer blocks consisting of several layers, including a simple linear layer, a feedforward network, and layers with self-attention. Self-attention enables a neural network to extract and use information from an unbounded context while maintaining the efficiency of processing non-sequential data.

4.1 Self-attention

The core concept of any attention mechanism is the ability to compare an item of interest with all surrounding items to gain a sense of contextual importance. To gain such insight, it is necessary to calculate the score for each item. Self-attention, discussed in this section, is an attention mechanism that relates different items of a single sequence to compute a representation of the sequence that reveals the relevance of a particular part of the sequence in the given context. Let us first define a query, key, and value embedding:

- **Query** - a current focus of attention that is being compared to all other preceding inputs,
- **Key** - a preceding input being compared to the current focus of attention (query),
- **Value** - a value used to compute the output for the current focus of attention (query).

Each of these three different roles is associated with a weight matrix, W^Q, W^K, and W^V for query, key, and value, respectively. If we denote (x_1, x_2, ... , x_n) as the sequence of input vectors (e.g., vector representations of the words), we can say that q_i, k_i, and v_i are role representations (query, key, value) of each of the input vectors:

$$q_i = W^Q x_i \textbf{;}\ k_i = W^K x_i \textbf{;}\ v_i = W^V x_i.$$

The score between query (current focus of attention x_i) and the key (element in the preceding context x_j) can be shown as:

$$score(x_i, x_j) = q_i \cdot k_j.$$

The problem with this setup is that the dot product between q_i and k_j can take arbitrarily large (positive or negative) values. To solve this problem, the scaled dot product is used,

which incorporates the dimensionality of the query and key vectors (which have the same size) into the calculation and scales the values as well. This approach is referred to as the scaled dot-product approach in the literature.

$$score(x_i, x_j) = \frac{q_i \cdot k_j}{\sqrt{d_k}}.$$

Similarly, in Section 3, where real-valued inputs x_1, x_2, ... , x_n can be represented by a single vector to benefit from highly parallelizable vector multiplication, here real-valued vector representations (e.g., of words) can be concatenated into a single matrix $\mathbf{X}$, again benefiting from matrix-matrix multiplication, which is easily parallelizable. Next, X is multiplied by the above weight matrices (W^Q, W^K, and W^V) to produce the matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ as follows:

$$Q = XW^Q;\ K = XW^K;\ V = XW^V.$$

Multiplying the Q and K^T matrices here yields **all requisite query-key comparisons**. After some additional steps to ensure learnable parameters, we obtain the final equation for computing self-attention:

$$SelfAttention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

When it comes to language modeling (i.e., predicting the next word in the sentence), it is not desirable to include words to be predicted in comparison, i.e., if we want to train the model to predict the next word based on this approach, the current query should not be compared to "future" words. To solve this issue, the QK^T matrix in the upper right triangle is set to zero, effectively denying information about the next word in the sentence. On the other hand, Devlin et al. [18] tackle the task of language modeling with an entirely new approach called **masked language modeling**. Masked language modeling and BERT are discussed in the following sections. To explore the transformer architecture (e.g., multi-head attention and transformer block) in more detail, the reader is advised to read the original work on transformers [14] and the literature used for this work, mainly [52] and [53].

4.2 BERT

When discussing language modeling for further use on downstream tasks, i.e. fine-tuning the pretrained language model, it is important to find the right objective for language modeling. In this light, Devlin et al. with "*BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*" [18] point out the shortcomings of left-to-right language modeling[5], such as work of Radford et al. [54]. The next paragraph briefly introduces transformer architecture and its connection with Generative pretrained transformer (GPT) and BERT models.

Transformer architecture, presented in the Figure 5, was originally implemented on the task of machine translation. When translating sentences, it is expected to have different

[5]Left-to-right language modeling objective is a task where an LM is trained to predict the next part of the sequence, given the previous parts (shortly discussed in Section 4)

lengths of sentences and unaligned words in the source language and target language. Machine translation is standardly tackled as a sequence-to-sequence (seq2seq) problem where the standard approach is the encoder-decoder architecture. Encoder-decoder consists of two main parts, an encoder that takes a variable-length input extracting useful information (features) that are then used as an input to decoder that conditioned on the encoded input through self-attention produces a variable-length output.

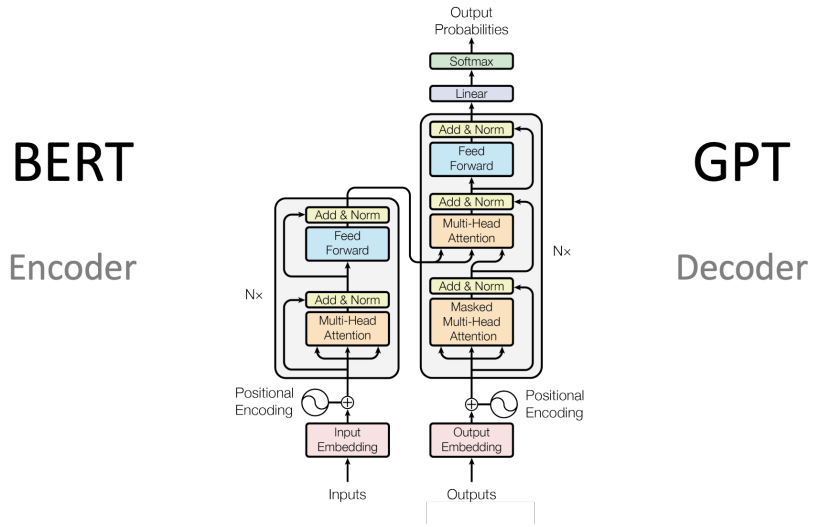

Figure 5: **Transformer architecture**: Architecture design used in the work of Vaswani et al. [14], consisting of a encoder and a decoder part, from which two models, BERT and GPT stem respectively. Adopted from work of Niklas Heidloff: *https://heidloff.net/article/foundation-models-transformers-bert-and-gpt/*

As shown in Figure 5, depicting original transformer encoder-decoder architecture, two famous models were created based on this architecture, the encoder only BERT [18] and the pure decoder GPT [54]. To abbreviate the topic, only the encoder of the transformer architecture is discussed. The encoder of a transformer consists of an arbitrary number of encoder blocks. Each encoder block starts with a self-attention layer, more specifically a multi-head self-attention that further facilitates the ability to encode multiple relationships and nuances for each part of the input, i.e., **token**[6]. The output of the multi-head self-attention then proceeds through position-wise feed-forward network (FFN) consisting of a linear layer, ReLU, and another linear layer. After each of the steps (multi-head self-attention and position-wise FFN), the residual connection is added along with the layer normalization. Figure 6 shows an encoder block previously described.

Following the original work of Vaswani et al. [14], BERT retains the encoder architecture with variations in the number of encoder blocks (L), hidden size (H), and self-attention heads (A). In the work, two main variants of BERT are presented: **BERT**$_{BASE}$ **(L=12, H=768, A=12)** and BERT$_{LARGE}$ (L=24, H=1024, A=16), the prior of which is used

[6]Token can be considered as a useful semantic unit for processing, common tokens are subword, word, and sentence tokens.

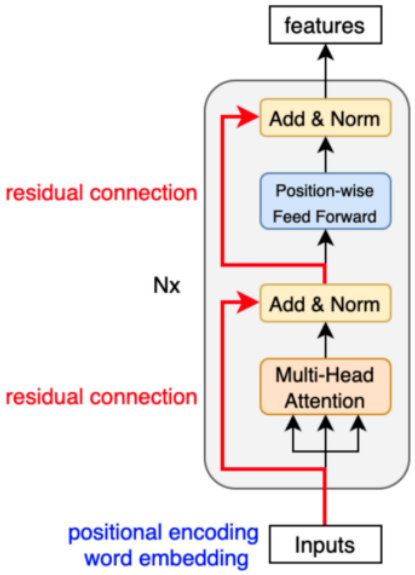

Figure 6: **Transformer encoder block**: Architecture used as an encoder originally in work of Vaswani et al. [14] and later in work of Devlin et al. [18]. Adopted from: *https://kikaben.com/transformers-encoder-decoder/*

in this thesis. To enable BERT to handle a variety of downstream tasks, such as text classification, relation extraction, sentiment analysis, and question answering, two special tokens are used in the input and output representations:

- [CLS] token - First token of every sequence. The final hidden state corresponding to this token is used as aggregate sequence representation for classification tasks.
- [SEP] token - Sentence separator, in case the input consists of two sentences, e.g. for question answering task.

For further detail on the input and output representation of BERT models, the reader is advised to read the work of Devlin et al. [18].

Authors split BERT pretraining into two separate tasks, **masked language modeling (MLM)** and next sentence prediction (NSP). As argued by authors, it is possible to gain more useful information, i.e. get better-contextualized representations when a single token is contextualized by the rest of the sequence (sentence), as compared to left-to-right or right-to-left model[7]. It is not desirable for a standard LM to attend to "future" tokens (words) in the sentence, as the training for predicting the next word in the sentence becomes trivial, as previously discussed in Section 4.1. In this light, Devlin et al. propose a new objective, that hides (**masks**) only a single part of the sequence that needs to be predicted, calling it

[7] "*Models where every token can only attend to previous tokens in the self-attention layers of the Transformer*" - [18]

MLM (masked language model)[8].

BERT model was **pretrained on BookCorpus [55] and English Wikipedia**.

4.3 SciBERT

Following the same design as above, Beltagy et al. [56] train the BERT model on scientific corpora to support further scientific-based use-cases. Of multiple pretrained models yielded in this work, the SciBERT with new SciVocab[9] WordPiece vocabulary uncased is used. New models that underwent pretraining were trained on **random sample of 1.14M papers from Semantic Scholar** [57].

[8]To mitigate the problems regarding special [MASK] token that does not appear during fine-tuning of the LM, extra steps in the objective are added.

[9]Original BERT uses WordPiece-based [12] vocabulary consisting of 30,000 tokens.

5 Datasets

In this Section, the datasets that were used to train the BERT and SciBERT models to classify relations are introduced. First, each of the datasets is defined and an example is provided, then the work of Shimorina et al. [58] is presented. Based on this work, the new dataset (**combo160**) is constructed, containing the relations of interest. Combo160 dateset is created through filtering and preprocessing aimed to use for training of relation extraction task with the OpenNRE [20] toolkit.

5.1 FewRel Dataset

Few-Shot Relation Classification Dataset (**FewRel**) by Han et al. [36] consists of 100 distinct relations, each accompanied by 700 instances. FewRel is created with the use of Wikipedia articles and Wikidata as a text corpus and the knowledge base[10] respectively. After the gathering of initial dataset through distant supervision, totaling in 122 relations and 122,000 instances, the dataset underwent human annotation filtering resulting in mentioned **70,000 instances**. A full list of relations, including their names and descriptions is available in the Appendix[11] of the paper. In Table2 below the examples of the FewRel data are listed:

ID	tokens	h	t
P26	[His, parents, are, Karl, von, Habsburg, and, Francesca, von, Habsburg, .]	"francesca von habsburg", Q1276954, [[7, 8, 9]]	"karl von habsburg", Q78515, [[3, 4, 5]]
P25	[Emmy, Acht\u00e9, was, the, mother, of, the, internationally, famous, opera, singers, Aino, Ackt\u00e9, and, Irma, Tervani, .]	"aino ackt\u00e9", Q259161, [[11, 12]]	"emmy acht\u00e9", Q4933685, [[0, 1]]

Table 2: **FewRel examples**: Head (**h**) and tail (**t**) are used interchangeably with subject and object annotation.

ID here stands for the unique property (relation) in Wikidata (e.g., P26 represents the "spouse" relation between two entities), it is followed by the tokenized sentence, after which the details of **h**ead and **t**ail[12] entities are given, i.e., surface form[13], unique item ID, and indexes referring to the tokenized list.

[10]Terms Knowledge Base (KB) and Knowledge Graph (KG) are often used interchangeably due to inconsistent or loose definitions.

[11]Not public due to details on the test data used for online evaluation. Available at: https://github.com/thunlp/FewRel/blob/master/data/pid2name.json

[12]Subject and object entities are sometimes considered head and tail entities in the literature.

[13]A raw, textual representation of an entity.

5.2 T-REx Dataset

Elsahar et al. [37] in "*T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples*" address the problem of small RE datasets by utilizing Wikidata[14] and DBpedia [59], a dataset consisting of Wikipedia abstracts, to form a new dataset called T-REx. T-REx contains **11 million triple alignments** from 6.2 million sentences. In this case, the triple alignment refers to the process of mapping the extracted entities from the natural language sentence with the triple in KG or KB, to form a distantly supervised training instance. For brevity, the examples from the T-REx dataset are not included.

5.3 DocRED Dataset

Compared to the previous two datasets, which were primarily developed for extracting sentence-level (intra-sentence) relations, Yao et al. [38] create a DocRED dataset for extracting document-level (inter-sentence) relations. DocRED is based on a similar design to T-REx and FewRel, in that the main and only data source is the aforementioned combination of Wikidata and Wikipedia abstracts through distant supervision. It is important to note that the tedious process of human annotation of part of the data using crowdsourcing has made a significant contribution to the field. The result is 5,053 human-annotated documents with 40,276 sentences and a total of **96 distinct relations** and a large distantly supervised dataset of 101,873 documents with 828,115 sentences.

5.4 WikiFact Dataset

Goodrich et al. [39] explore new metrics for evaluating the factual accuracy of the generated text, primarily for the RE task. Similar to the previously presented datasets, under the distant supervision assumption [60], a new dataset based on Wikidata and Wikipedia, WikiFact, is created. WikiFact consists of two distinct parts based on the training objective, data for the relation classifier (relation extraction) and data for fact extraction (paragraph and sentence based). In this work, only the data for RE is used, which is 13 GB in total. Table 3 previews examples from the WikiFact dataset. Compared to the structure of the data in FewRel2, the authors mark the entities in untokenized sentences, which leaves room for task-specific preprocessing.

target	inputs	subject	object
P0	SUBJ{Art Nalls} was born in 1954 in OBJ{Alexandria}, Virginia just outside Washington, D.C. and grew up in that area.	Art Nalls	Alexandria
P0	SUBJ{Cerner} and executives at Girard agreed that Girard did not have adequate staff to manage the acquisition and OBJ{implementation} of the system.	Cerner	implementation

Table 3: **WikiFact examples**: Inputs (sentences) have annotated subject (SUBJ{ }) and object (OBJ{ }) spans.

[14] https://www.wikidata.org/

5.5 Wiki20m Dataset

Comparable to the previously covered datasets, Wiki20m [24] utilizes distant supervision via Wikipedia and Wikidata. Wiki20 is originally constructed for bag-level relation extraction, a task that aims to spread relation extraction on multiple sentences (i.e. bag of sentences). Wiki20m is the version of Wiki20 with a manually annotated test set. Each instance in the dataset resembles the structure of FewRel with tokenized sentence, head and tail data, and relation label.

5.6 WebRED Dataset

In "*WebRED: Effective Pretraining And Finetuning For Relation Extraction On The Web*" [40] Ormandi et al. point out generalization problems that concern Wikipedia-Wikidata[15] trained models, since the text on Wikipedia follows a certain structure and certain constraints. To mitigate generalization problems, a set of web domains with high linguistic quality and factually correct content were ranked by human annotators. From the selected web domains, the large corpus was created and linked to Wikidata triples in a process similar to previous datasets. Then, the part of the data was subjected to a human annotation process similar to DocRED. The result is a two-part dataset with **523 unique (Wikidata) relations**, with the **117,717 human annotated** and **199,786,781 weakly supervised** examples.

5.7 Relation Extraction Database Based on Wikidata

While the datasets presented were created for seemingly different tasks, they share the knowledge base for defining and extracting relations, Wikidata. With this property in mind, Shimorina et al. [58] design a sentence-level RE database based on the aforementioned datasets (FewRel, T-REx, DocRED, WikiFact, Wiki20m, WebRED). First, the datasets are preprocessed, including deduplication and extraction of designated entities from unwanted formats. Then, the instances (sentences) or rather, relations are labeled in a uniform way, where OBJ and SUBJ represent the entities involved in a given relation. Results are **47,390,557** instances across **1,022 unique relations**, including "**P0**" (no relation) and "**NA**" (unknown relation).

[15] Relation extraction datasets constructed from Wikipedia and Wikidata through distant supervision.

6 Experimental setup

6.1 Combo160 Dataset

Since this work is concerned with the scientific domain, it is desirable to train the models on data that exhibit scientific relationships. To obtain such relations, the Wikidata list of properties in the science domain[16] was examined. Particular attention was paid to relations in chemistry, physics, biology, mathematics, geology, and astronomy, resulting in 341 applicable relations (see Appendix A). Based on these relations, the database created by Shimorina et al. [58] was queried to filter out the ones of interest, resulting in a new dataset with 176 relations. The number of instances for each relation is limited to 10,000 to prevent further increase in disproportionality of the data, since the majority of the examples (66%) belong to the classes "P0" (no relation) and "NA" (unknown relation).

The combo160 dataset is then exported to csv format for further preprocessing using python with appropriate libraries (pandas [61,62], scikit-learn [63]). First, the empty values and notation SUBJ and OBJ are cleaned up in the records. Clean sentences are then tokenized using the BasicTokenizer implemented in OpenNRE to make the input conform to the Toolkit standards. Following tokenization, low frequency relations (less than 3 times) are removed to allow splitting between training, test, and validation, with each relation occurring at least once per set. Using scikit-learn function `train_test_split()` with stratification based on relation ID (e.g., P1234), the dataset is split into 80:15:5 ratio to train, test, and validation subsets respectively. After the preprocessing, the dataset is left with **161 unique relations** (Appendix B), with **301,062 examples** in total. Hence we name the dataset combo160. In the Table 4 below, summary of datasets information is given with respect to the work of Shimorina et al [58] [17].

dataset	# instances	# R	% neg.
FewRel	56,000	80	0%
T-REx	12,081,023	652	0%
DocRED	778,914	96	0%
WikiFact	33,628,338	934	92%
Wiki20m	738,463	81	60%
WebRED	107,819	385	54%
Unified database	47,390,557	1,022	66%
Combo160	**301,062**	**161**	**6.6%**

Table 4: **Dataset summary**: with the number of instances, the number of relations, and negative relations.

Table 4 shows the total number of instances (# instances), relations (# R), and percentage of negative relations (% neg.) in each of the datasets. Relations "P0" (no relation) and "NA" (unknown relation) are considered negative in this context. Negative examples can be excluded if a binary classification (yes/no relation between entities) is performed before

[16] https://www.wikidata.org/wiki/Wikidata:List_of_properties/science

[17] Dataset statistics presented in Table 4 refer to sentence-level RE and are calculated after preprocessing procedures, such as deduplication.

relation classification (RC). Essentially, the RE is further divided into two approaches: the first, binary classification before RC and the second, RC with the classes "no relation" and/or "unknown" relation classes. This work treats "P0" and "NA" as regular relations although, usually, most entity co-occurrences are either undefined or not-relation, which is the main rationale behind high percentage of negative examples in datasets such as WikiFact, Wiki20m, WebRED and the unified database [58]. The exploration of the impact of the negative examples ratio is left for future work.

6.2 OpenNRE toolkit

OpenNRE is an open source and extensible toolkit that provides a unified framework for implementing relation extraction models, introduced with the work of Han et al. [20] "*OpenNRE: An Open and Extensible Toolkit for Neural Relation Extraction*". The authors point out the need for such a toolkit because most research on RE implements its work for a specific setup, making the comparison, re-implementation, variation, deployment, and evaluation a relatively difficult task. The authors also address the problem of code reusability by providing extensible base implementations for most tasks that precede or follow RE, such as tokenization (word and subword level), common neural layers, encoder module, data processing, model training and evaluation. OpenNRE also allows 3 approaches to RE:

- **Sentence-level RE** (RE from sentence, i.e. only the existence of relations inside a single sentence is assumed),
- **Bag-level RE** (RE from multiple sentences, i.e. relations can exist across multiple sentences that appear consecutively),
- **Document-level RE** (RE from the whole document).

In this work, OpenNRE is used due to its extensibility and ease of use, as the toolkit contains a complete procedure for training the BERT model for RE following the work of Soares et al. [64].

6.3 Training Setup

Soares et al. with "*Matching the Blanks: Distributional Similarity for Relation Learning*" investigate the capabilities of BERT in extracting relations (classification) and then train the BERT model on new objective specifically designed to capture relation representation, named matching the blanks (MTB). In this work, we investigate one of the presented relation classification fine-tuning procedures. Let us first define a sequence of tokens (x), e.g. words, as $x = [x_0, x_1, ..., x_n]$, where, similarly to original setup, $x_0 = [CLS]$ and $x_n = [SEP]$ are special start and end markers. Moreover, let $s_1 = (i, j)$ and $s_2 = (k, l)$ be pairs of integers such that $0 < i < j - 1, j < k, k \leq l - 1$, and $l \leq n$. Here, relation (r) is represented as $r = (x, s_1, s_2)$, where s_i represents the entity mentions in the sentence.

In this thesis, BERT models are fine-tuned for RC with **entity marker** tokens that incorporate s_1 and s_2 entity spans into the input via special tokens: [E1$_{start}$], [E1$_{end}$],

[E2$_{start}$], and [E2$_{end}$]. Resulting in an augmented sequence ($\tilde{x}$) as:

$$\tilde{x} = \big[x_0, ..., [E1_{start}], x_i, ..., x_{j-1}, [E1_{end}], ..., [E2_{start}], x_k, ..., x_{l-1}, [E2_{end}], ..., x_n\big].$$

Let us preview this in the exemplary sentence:
If sentence (x) is:

$$x = \big[\boldsymbol{he}, is, seen, as, one, of, the, founders, of, modern, \boldsymbol{archeology}, in, czech, lands, "."\big];$$

and entity spans s_1 and s_2 are:

$$\boldsymbol{s_1} = (0, 0) \text{ and } \boldsymbol{s_2} = (10, 10);$$

then augmented sentence ($\tilde{x}$) is:

$$\tilde{x} =$$
$$\big[[\boldsymbol{E1_{start}}], he, [\boldsymbol{E1_{end}}], is, ..., modern, [\boldsymbol{E2_{start}}], archeology, [\boldsymbol{E2_{end}}], in, czech, lands, "."\big].$$

With this set-up, two models are fine-tuned with the combo160 dataset on the RC task with parameters set up as elaborated in Table 5. To enable training for the task of multi-label classification, outputs of the encoders (BERT and SciBERT) are forwarded to a neural network consisting of a linear layer, dropout layer, and softmax layer to output the probability distribution over the number of classes.

Model	BERT$_{base}$ uncased / SciBERT SciVocab uncased
Dataset	Combo160
Pooler	Entity
Entity masker	No
Batch size	8
Learning rate	2e-5
Maximum length	128
Maximum epochs	3
Seed	42
Optimizer	Adamw [65]

Table 5: **Training parameters set-up**

Each of the two models (BERT and SciBERT) was trained for 3 epochs with a total of $\sim$ 12 hours of training on an Ubuntu 20.04 machine with a single NVIDIA GeForce GTX 1050 Mobile (3GB) GPU and Intel Core i7-7700HQ CPU. The results of both BERT$_{base}$ uncased and SciBERT SciVocab uncased models are compared following an identical training set-up to support the arguments discussed. It is important to note that BERT$_{base}$ and SciBERT have the same architecture and differ only in the corpora used to pretrain the model.

7 Results

In this Section, the results of the thesis are presented. First, the inference of two models (BERT and SciBERT) is evaluated with standard multiclass classification metrics such as accuracy, precision and F1-score with micro and macro averaging. Then, the significance of the results is discussed and concluded with examples of the raw unannotated data.

Let us first consider the distribution of relations in the training data (Figure 7) and the test data (Figure 8). Comparing the top 20 relations (by the number of instances) in both datasets, we obtain a significant overlap with a difference in one relation P706 (in train) versus P279 (in test). The top 20 include relations such as: *location* (P276), *father* (P22), *sibling* (P3373), *instance of* (P31), *located in the administrative territorial entity* (P131), *owned by* (P127), *field of work* (P101), and of course the negative classes *unknown* (NA /PNAN) and *no relation* (P0). The above relations portray general terms rather than relations that are characteristic of scientific work. On the other hand, relations such as *chromosome* (P1057), *monomer of* (P4599), *pathogen transmission process* (P1060), *lymphatic drainage* (P2288), *research site* (P6153), *decreased expression in* (P1910), and *inflorescence* (P3739), which might be considered more prone to scientific work, appear in the bottom 20 places (by occurrence). Arguably, it is expected to have a higher overall occurrence of "general" relations, compared to "scientific domain" relations in the dataset constructed mainly from Wikipedia[18].

With this in mind, we move on to the results of two models reported in Table 6.

	BERT	SciBERT
Accuracy	**0.998925725881054**	0.9987707454750053
Micro precision	**0.9236978621443692**	0.914232253537598
Micro recall	**0.9093136693613025**	0.8951382050518698
Micro F1-score	**0.9164493272093612**	0.9045844808034992
Macro precision	0.7452305633653035	**0.7744999417648366**
Macro recall	0.7015722908580357	**0.7263660522886056**
Macro F1-score	0.7164054666737936	**0.7411929693031761**

Table 6: **The results for BERT and SciBERT**: in terms of accuracy, micro and macro averaged precision, recall and F1-score.

The first metric discussed, **accuracy**, yields somewhat similar results, with a performance difference of $1.54980406 \cdot 10^{-4}$ in favor of BERT. If we revisit the definition of accuracy (Section 2.1.1), it becomes clear that this metric only rewards correct predictions and attenuates the proportion of correctness per class. While intuitive, this metric yields biased results when the dataset is unbalanced, as is the case with Combo160 (Figures 7 and 8). Meaning, the model that more accurately predicts dominant classes (presumably BERT) could have a better result as rated by the accuracy metric.

To discuss the matter further, let us explore micro and macro averaging in greater depth. According to Grandini et al. [32] the idea of micro-averaging is to consider all the units together, without considering disproportion between classes.

[18]Reader is advised to explore Appendices B and C containing further information on relations in the dataset.

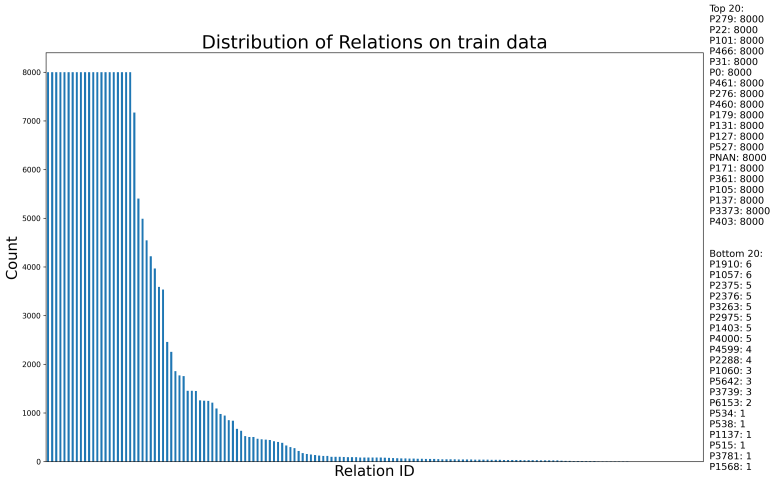

Figure 7: **Relation distribution in train data**

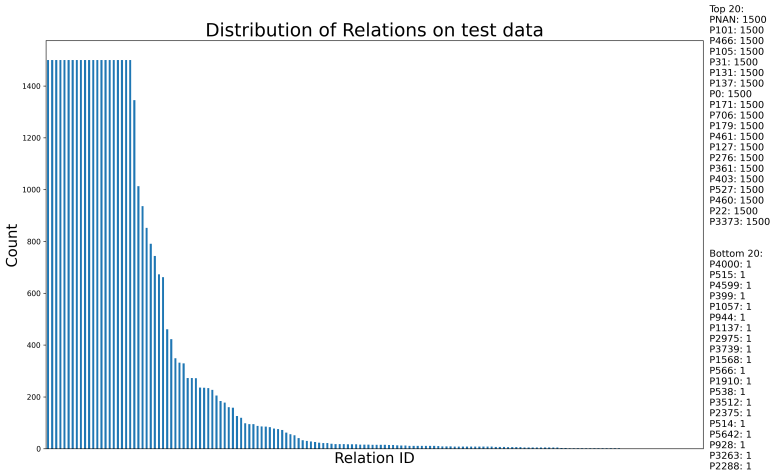

Figure 8: **Relation distribution in test data**

Similar to accuracy, BERT shows negligibly better performance in **micro-averaged precision** with $9.465608607 \cdot 10^{-3}$ advantage. This advantage is also to be expected since micro-averaging first sums all units, i.e., TP and FP, and then calculates precision based on the sums, again neglecting the inequality of classes in the dataset. The comparison of **micro-averaged recall** yields similar results with a more significant advantage of $1.417546431 \cdot 10^{-2}$ in the performance of BERT. **Micro F1-score**, as a harmonic mean between (micro) precision and recall, gives a better assessment of the overall performance of the model than the two previously mentioned metrics. Looking at the results of micro F1-score, a repeating trend is seen when discussing micro-averaged metrics, with BERT performing with a lead of $1.186484641 \cdot 10^{-2}$. **Thus, based on the micro-averaged**

metrics, it can be concluded that BERT performs better with the more dominant classes as compared to SciBERT.

Looking at the macro-averaged metrics, there is a significant decline in performance scores in general. This behaviour can be explained by the unbalanced dataset, as macro-averaged metrics tend to neglect the correlation between class size and overall scores [32]. This means that macro-averaged metrics, by virtue of their calculations, have the great side effect of giving equal importance to each class (regardless of the number of instances in the class). Let us now consider the macro-averaged scores of the models. A significant difference is found when comparing **macro precision** of two models, as SciBERT scores $2.92693784 \cdot 10^{-2}$ better here. This means that the SciBERT model classifies the relations more confidently. Comparable results are also manifested in the **macro recall** and **macro F1-score**.

Two important conclusions can be drawn from the analysis of the provided metrics measured on SciBERT and BERT models in classifying relations:

- **The BERT model performs better on more dominant classes overall (better results on micro-averaged metrics and accuracy) and, in particular, correctly classifies examples of dominant relations (greater macro recall).**
- **The SciBERT model is competitive with BERT when it comes to more dominant relations, and shows better results on less dominant classes (better overall results on macro-averaged metrics).**

As noted at the beginning of this section, the dominant (frequent) relations tend to be more "general", while the less dominant (infrequent) relations tend to be more "scientific". Given this, it can be concluded that **SciBERT performs better when presented with relations from the scientific domain**.

8 Conclusion

This master thesis first introduces the field of NLP with its history and the NLP task, **relation extraction**. Relation extraction, as a subtask of information extraction, attempts to automatically extract the relations between actors in the form of a triple (subject, relation, object) from texts. This thesis is concerned with the training (fine-tuning) of sentence-level relation extraction models, focusing mainly on relation classification as a multi-class classification problem. For this task, the pretrained transformer-encoder models BERT and SciBERT are used and compared to observe the effects of the pretrained corpora.

Throughout the thesis, the reader is introduced to basic concepts related to neural networks (NN), transformer architecture, pretraining, language modeling, and masked language modeling (MLM). Relevant datasets suitable for the task are explored and experimented with, culminating in the construction of a new dataset, **combo160**. Two models, SciBERT and BERT, are then trained on this newly constructed dataset with 161 relation types - combo160.

Finally, the results of the two models are presented and compared in order to draw conclusions about the pretraining corpora used. This work tests the hypothesis: "**Will the SciBERT model perform better than the BERT model on the relation extraction task on scientific corpora?**". Using relevant metrics for classification tasks, such as accuracy and (micro- and macro-) averaged precision, recall and F1-score, it is shown that BERT performs marginally better on accuracy and micro-averaged metrics. This implies better performance on more dominant classes (which turn out to be less "scientific domain" and more "general"), while **SciBERT model outperforms the BERT model when it comes to relations specific to the scientific domain**, implied by better results on the macro-averaged metrics that account for class disproportions in the calculation. To conclude, we list two important contributions of this work:

- Creation (selection) of the new scientifically oriented dataset useful for further benchmarks on reletion extraction- **combo160**;
- Two models (BERT and SciBERT) fine-tuned for RE available for use through **OpenNRE toolkit** [20];

To further support the conclusion, the issue of defining the term "scientific" relationship needs to be addressed in more detail. In addition, other models with different pretraining objectives and architectures should be explored and evaluated against the combo160 dataset. The sole dataset, combo160, should be further analyzed to reduce and find the optimal and necessary number of relations. This and the study of the effects of relation distribution (especially negative relations) are the subject of future work.

9 References

[1] "What is a natural language?." Accessed on June 15, 2023, https://www.dictionary.com/browse/natural-language.

[2] J. Eisenstein, *Introduction to Natural Language Processing (Adaptive Computation and Machine Learning series)*. MIT Press, 2019.

[3] W. J. Hutchins, "The georgetown-ibm experiment demonstrated in january 1954," in *Machine Translation: From Real Users to Research* (R. E. Frederking and K. B. Taylor, eds.), (Berlin, Heidelberg), pp. 102–114, Springer Berlin Heidelberg, 2004.

[4] J. Hutchins, "From first conception to first demonstration: the nascent years of machine translation, 1947–1954. a chronology," *Machine Translation*, vol. 12, pp. 195–252, 09 1997.

[5] T. Winograd, "Understanding natural language," *Cognitive Psychology*, vol. 3, no. 1, pp. 1–191, 1972.

[6] J. Weizenbaum, "Eliza—a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, p. 36–45, jan 1966.

[7] R. A. Harris, *The Linguistics Wars*. Oxford University Press USA, January 1993.

[8] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts: The MIT Press, May 1999.

[9] J. Hirschberg and C. Manning, "Advances in natural language processing," *Science (New York, N.Y.)*, vol. 349, pp. 261–266, 07 2015.

[10] D. T. Team, "History and present of natural language processing." Deep Talk AI, 2023. Accessed on June 14, 2023.

[11] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2016.

[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016.

[13] B. Lorica and M. Loukides, "What machine learning means for software development," *O'Reilly Radar*, 2018. Accessed on June 14, 2023.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[15] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[16] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018.

[17] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," 2018.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[19] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, "A survey of large language models," 2023.

[20] X. Han, T. Gao, Y. Yao, D. Ye, Z. Liu, and M. Sun, "OpenNRE: An open and extensible toolkit for neural relation extraction," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, (Hong Kong, China), pp. 169–174, Association for Computational Linguistics, Nov. 2019.

[21] S. Sarawagi, "Information extraction," *Foundations and Trends® in Databases*, vol. 1, no. 3, pp. 261–377, 2008.

[22] S. Pawar, G. K. Palshikar, and P. Bhattacharyya, "Relation extraction : A survey," 2017.

[23] E. Bassignana and B. Plank, "What do you mean by relation extraction? a survey on datasets and study on scientific relation classification," 2022.

[24] X. Han, T. Gao, Y. Lin, H. Peng, Y. Yang, C. Xiao, Z. Liu, P. Li, J. Zhou, and M. Sun, "More data, more relations, more context and more openness: A review and outlook for relation extraction," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, (Suzhou, China), pp. 745–758, Association for Computational Linguistics, Dec. 2020.

[25] A. Poleksić, "Paralelizacija izračuna dubokih neuralnih mreža na grafičkim procesorima," 2021.

[26] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, (Denver, Colorado), pp. 39–48, Association for Computational Linguistics, June 2015.

[27] C. Alt, M. Hübner, and L. Hennig, "Improving relation extraction by pre-trained language representations," 2019.

[28] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Berlin, Germany), pp. 207–212, Association for Computational Linguistics, Aug. 2016.

[29] P.-L. Huguet Cabot and R. Navigli, "REBEL: Relation extraction by end-to-end language generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, (Punta Cana, Dominican Republic), pp. 2370–2381, Association for Computational Linguistics, Nov. 2021.

[30] W. Tang, B. Xu, Y. Zhao, Z. Mao, Y. Liu, Y. Liao, and H. Xie, "UniRel: Unified representation and interaction for joint relational triple extraction," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, (Abu Dhabi, United Arab Emirates), pp. 7087–7099, Association for Computational Linguistics, Dec. 2022.

[31] B. Taillé, V. Guigue, G. Scoutheeten, and P. Gallinari, "Let's Stop Incorrect Comparisons in End-to-end Relation Extraction!," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 3689–3701, Association for Computational Linguistics, Nov. 2020.

[32] M. Grandini, E. Bagli, and G. Visani, "Metrics for multi-class classification: an overview," 2020.

[33] "Cambridge dictionary: Knowledge." Accessed on June 16, 2023, https://dictionary.cambridge.org/dictionary/english/knowledge.

[34] A. Hogan, E. Blomqvist, M. Cochez, C. D'amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," *ACM Computing Surveys*, vol. 54, pp. 1–37, jul 2021.

[35] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, pp. 494–514, feb 2022.

[36] X. Han, H. Zhu, P. Yu, Z. Wang, Y. Yao, Z. Liu, and M. Sun, "FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 4803–4809, Association for Computational Linguistics, Oct.-Nov. 2018.

[37] H. Elsahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, F. Laforest, and E. Simperl, "T-REx: A large scale alignment of natural language with knowledge base triples," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.

[38] Y. Yao, D. Ye, P. Li, X. Han, Y. Lin, Z. Liu, Z. Liu, L. Huang, J. Zhou, and M. Sun, "DocRED: A large-scale document-level relation extraction dataset," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 764–777, Association for Computational Linguistics, July 2019.

[39] B. Goodrich, V. Rao, P. J. Liu, and M. Saleh, "Assessing the factual accuracy of generated text," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, (New York, NY, USA), p. 166–175, Association for Computing Machinery, 2019.

[40] R. Ormandi, M. Saleh, E. Winter, and V. Rao, "Webred: Effective pretraining and finetuning for relation extraction on the web," 2021.

[41] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, (New York, NY, USA), p. 1247–1250, Association for Computing Machinery, 2008.

[42] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, p. 78–85, sep 2014.

[43] A. Pingle, A. Piplai, S. Mittal, A. Joshi, J. Holt, and R. Zak, "Relext: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '19, (New York, NY, USA), p. 879–886, Association for Computing Machinery, 2020.

[44] H. Yu, H. Li, D. Mao, and Q. Cai, "A relationship extraction method for domain knowledge graph construction," *World Wide Web*, vol. 23, pp. 735–753, March 2020.

[45] F.-L. Li, H. Chen, G. Xu, T. Qiu, F. Ji, J. Zhang, and H. Chen, "Alimekg: Domain knowledge graph construction and application in e-commerce," CIKM '20, (New York, NY, USA), p. 2581–2588, Association for Computing Machinery, 2020.

[46] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, "Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction," 2018.

[47] F. Rosenblatt, "The perceptron - a perceiving and recognizing automaton," Tech. Rep. 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York, January 1957.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[49] A. Veit, M. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," 2016.

[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[51] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1724–1734, Association for Computational Linguistics, Oct. 2014.

[52] D. Jurafsky and J. H. Martin, *Speech and Language Processing (3rd ed. draft)*. 2023. Unpublished: Accessed on June 22, 2023.

[53] F. Chollet, *Deep Learning with Python, Second Edition*. Manning Publications, November 2021.

[54] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[55] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," 2015.

[56] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, (Hong Kong, China), pp. 3615–3620, Association for Computational Linguistics, Nov. 2019.

[57] W. Ammar, D. Groeneveld, C. Bhagavatula, I. Beltagy, M. Crawford, D. Downey, J. Dunkelberger, A. Elgohary, S. Feldman, V. Ha, R. Kinney, S. Kohlmeier, K. Lo, T. Murray, H.-H. Ooi, M. Peters, J. Power, S. Skjonsberg, L. L. Wang, C. Wilhelm, Z. Yuan, M. van Zuylen, and O. Etzioni, "Construction of the literature graph in semantic scholar," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, (New Orleans - Louisiana), pp. 84–91, Association for Computational Linguistics, June 2018.

[58] A. Shimorina, J. Heinecke, and F. Herledan, "Knowledge extraction from texts based on Wikidata," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, (Hybrid: Seattle, Washington + Online), pp. 297–304, Association for Computational Linguistics, July 2022.

[59] M. Brümmer, M. Dojchinovski, and S. Hellmann, "DBpedia abstracts: A large-scale, open, multilingual NLP training corpus," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, (Portorož, Slovenia), pp. 3339–3343, European Language Resources Association (ELRA), May 2016.

[60] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural*

Language Processing of the AFNLP, (Suntec, Singapore), pp. 1003–1011, Association for Computational Linguistics, Aug. 2009.

[61] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020.

[62] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

[63] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[64] L. Baldini Soares, N. FitzGerald, J. Ling, and T. Kwiatkowski, "Matching the blanks: Distributional similarity for relation learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 2895–2905, Association for Computational Linguistics, July 2019.

[65] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

List of Tables

List of Figures

Appendices

A List of desired relations

NA P0 P10019 P101 P1016 P10176 P10228 P10241 P1033 P10374 P1046 P105 P1050 P10568 P1057 P1060 P10601 P10602 P10614 P10630 P1074 P10906 P10946 P11220 P11231 P1137 P1145 P11484 P1171 P11801 P11814 P1199 P121 P1227 P126 P127 P128 P129 P131 P1312 P1322 P1349 P137 P1382 P1403 P141 P1420 P1425 P1478 P1479 P1531 P1535 P1542 P1560 P1568 P1571 P1582 P1604 P1605 P1606 P1647 P1660 P1672 P1677 P1678 P1703 P1704 P171 P179 P183 P1830 P1851 P1853 P1889 P1903 P1909 P1910 P1911 P1912 P1913 P1914 P1915 P1916 P1917 P1918 P1924 P196 P1990 P1995 P2061 P2094 P2152 P2155 P2156 P2159 P2175 P2176 P22 P2239 P2283 P2286 P2288 P2289 P2293 P2329 P2375 P2376 P2384 P2396 P2414 P2433 P2548 P2571 P2575 P2577 P2597 P2695 P2743 P276 P2784 P2789 P279 P2820 P2827 P2839 P2841 P2849 P2959 P2974 P2975 P3094 P31 P3137 P3179 P3189 P3190 P3205 P3261 P3262 P3263 P3264 P3310 P3354 P3355 P3356 P3357 P3358 P3359 P3364 P3373 P3403 P3432 P3433 P3464 P3489 P3490 P3491 P3493 P3512 P3578 P3592 P361 P3648 P366 P3739 P3741 P376 P3771 P3772 P3773 P3774 P3775 P3776 P3777 P3778 P3779 P3780 P3781 P3815 P397 P398 P399 P4000 P403 P4044 P405 P414 P4147 P4149 P427 P4425 P4426 P4545 P4552 P4599 P460 P4600 P461 P4628 P466 P4743 P4770 P4774 P4777 P4843 P4844 P4873 P4875 P4882 P4915 P4952 P4954 P5040 P5041 P5042 P5095 P5131 P5132 P5133 P5134 P5135 P5136 P514 P515 P5166 P517 P522 P523 P5236 P524 P5248 P527 P5304 P534 P537 P538 P5386 P5446 P556 P5572 P5588 P5589 P5607 P5642 P565 P566 P567 P568 P579 P5841 P589 P59 P6099 P61 P6153 P6185 P6212 P6259 P629 P636 P65 P660 P680 P6803 P681 P682 P684 P6855 P6875 P688 P6884 P689 P690 P693 P694 P697 P702 P703 P706 P720 P7309 P744 P7442 P7479 P7500 P7582 P7603 P769 P770 P780 P783 P784 P785 P786 P787 P788 P789 P8005 P8026 P8045 P816 P8193 P8194 P828 P8324 P8329 P8339 P8363 P8403 P8450 P8459 P859 P873 P8789 P881 P8824 P8864 P8865 P8866 P8872 P9030 P9072 P913 P922 P923 P9235 P924 P925 P926 P927 P9275 P928 P9353 P944 P9566 P970 P9714 P9745 P9831 P9888 P9977

B Relations in combo160

RID	name	RID	name
P279	subclass of	P556	crystal system
P22	father	P3179	territory overlaps
P105	taxon rank	P636	route of administration
P61	discoverer or inventor	P2695	type locality (geology)
P137	operator	P2384	statement describes
P366	use	P3780	active ingredient in
NA	unknown	P427	taxonomic type
P403	mouth of the watercourse	P3094	develops from
P706	located on terrain feature	P3491	muscle insertion
P59	constellation	P780	symptoms
P361	part of	P769	significant drug interaction
P171	parent taxon	P703	found in taxon
P3373	sibling	P1479	has contributing factor
P398	child astronomical body	P568	overlies
P922	magnetic ordering	P816	decays to
P127	owned by	P1990	species kept
P65	site of astronomical discovery	P926	postsynaptic connection
P131	located in the administrative territorial entity	P1851	input set
P179	part of the series	P720	asteroid spectral type
P1889	different from	P744	asteroid family
P527	has part	P2152	antiparticle
P2175	medical condition treated	P1531	"parent of this hybrid
P460	said to be the same as	P514	interleaves with
P397	parent astronomical body	P3262	has anatomical branch
P1995	health specialty	P1074	fictional analog of
P276	location	P522	type of orbit
P2597	Gram staining	P2839	gait
P461	opposite of	P3261	anatomical branch of
P2176	drug used for treatment	P702	encoded by
P0	no_relation	P523	temporal range start
P31	instance of	P376	located on astronomical location
P126	maintained by	P684	ortholog
P1605	has natural reservoir	P567	underlies
P466	occupant	P924	possible treatment
P629	edition or translation of	P128	regulates (molecular biology)
P2959	permanent duplicated item	P399	companion of
P101	field of work	P923	medical examinations
P2094	competition class	P517	interaction
P4552	mountain range	P2974	habitat
P121	item operated	P2575	measures
P789	edibility	P2289	venous drainage
P129	physically interacts with	P783	hymenium type
P688	encodes	P3205	patient of
P2329	antagonist muscle	P3190	innervates
P183	endemic to	P4743	animal breed
P3403	coextensive with	P3815	volcano observatory
P414	stock exchange	P2743	this zoological name is coordinate with
P1542	has effect	P1571	codomain
P1582	natural product of taxon	P2414	substrate of
P196	minor planet group	P1057	chromosome
P376	located on astronomical body	P2159	computes solution to
P3189	innervated by	P913	notation
P1050	medical condition	P1678	has vertex figure
P2789	connects with	P3490	muscle origin
P828	has cause	P2293	genetic association
P1830	owner of	P5135	greater than
P1924	vaccine for	P2288	lymphatic drainage
P881	type of variable star	P1916	gene substitution association with
P141	IUCN conservation status	P4599	monomer of
P2283	uses	P3512	means of locomotion
P2155	solid solution series with	P1910	decreased expression in
P1672	this taxon is source of	P2375	has superpartner
P859	sponsor	P1046	discovery method
P1322	dual to	P928	activating neurotransmitter
P1420	taxon synonym	P4600	polymer of
P770	cause of destruction	P566	basionym
P3137	parent peak	P944	Code of nomenclature
P524	temporal range end	P2975	host
P2849	produced by	P1403	original combination
P689	afflicts	P5642	risk factor
P681	cell component	P1060	pathogen transmission process
P927	anatomical location	P3263	base
P1535	used by	P4000	has fruit type
P2286	arterial supply	P2376	superpartner of
P682	biological process	P534	streak color
P1478	has immediate cause	P6153	research site
P788	mushroom ecological type	P538	fracturing
P1312	has facet polytope	P1137	fossil found in this unit
P1382	partially coincident with	P3739	inflorescence
P3781	has active ingredient	P515	phase of matter
P1568	definition domain		

Table 7: **Combo160 relations**

C Relation counts in combo160

RID	Train	Test	RID	Train	Test	RID	Train	Test
PNAN	8000	1500	P4599	4	1	P2175	1454	273
P944	8	1	P4552	7170	1345	P2159	24	5
P928	8	1	P427	332	62	P2155	82	15
P927	507	95	P414	2458	461	P2152	43	8
P926	11	2	P403	8000	1500	P2094	4218	791
P924	32	6	P4000	5	1	P1995	979	184
P923	59	11	P399	8	1	P1990	66	12
P922	25	5	P398	4990	936	P196	445	83
P913	34	6	P397	5402	1013	P1924	47	9
P881	90	17	P3815	28	5	P1916	30	5
P859	452	85	P3781	1	2	P1910	6	1
P828	1758	329	P3780	55	11	P1889	4545	852
P816	90	17	P376	2253	423	P1851	27	5
P789	38	8	P3739	3	1	P1830	633	119
P788	55	11	P366	3968	744	P183	1249	234
P783	11	2	P361	8000	1500	P179	8000	1500
P780	277	52	P3512	9	1	P171	8000	1500
P770	174	33	P3491	46	8	P1678	29	5
P769	69	13	P3490	15	3	P1672	1258	236
P744	38	7	P3403	402	76	P1605	40	8
P720	136	26	P3373	8000	1500	P1582	1250	235
P706	8000	1500	P3263	5	1	P1571	14	2
P703	64	12	P3262	45	8	P1568	1	1
P702	34	6	P3261	47	9	P1542	1858	349
P689	385	72	P3205	12	2	P1535	525	98
P688	147	28	P3190	78	14	P1531	44	8
P684	48	9	P3189	81	15	P1479	90	17
P682	507	95	P3179	34	7	P1478	84	16
P681	469	88	P3137	418	78	P1420	82	15
P65	672	126	P31	8000	1500	P141	1210	227
P636	39	8	P3094	58	11	P1403	5	2
P629	297	56	P2975	5	1	P1382	82	15
P6153	2	2	P2974	24	5	P137	8000	1500
P61	3533	662	P2959	61	11	P1322	117	22
P59	3589	673	P2849	62	11	P1312	158	30
P568	98	18	P2839	33	6	P131	8000	1500
P567	99	19	P279	8000	1500	P129	946	178
P566	10	1	P2789	1450	272	P128	37	7
P5642	3	1	P276	8000	1500	P127	8000	1500
P556	217	41	P2743	72	14	P126	851	160
P538	1	1	P2695	85	16	P121	1456	273
P534	1	2	P2597	1770	332	P1137	1	1
P527	8000	1500	P2575	40	8	P1074	51	10
P524	96	18	P2414	30	5	P1060	3	2
P523	117	22	P2384	17	3	P1057	6	1
P522	123	23	P2376	5	2	P1050	841	158
P517	98	18	P2375	5	1	P105	8000	1500
P515	1	1	P2329	39	8	P1046	11	2
P514	7	1	P2293	54	11	P101	8000	1500
P5135	10	2	P2289	42	8	P0	8000	1500
P4743	24	5	P2288	4	1			
P466	8000	1500	P2286	84	16			
P461	8000	1500	P2283	460	86			
P4600	9	2	P22	8000	1500			
P460	8000	1500	P2176	1092	205			

Table 8: **Combo160 relations train-test counts**