

Masters Thesis in Computer Science

Extending Semantic Hypergraphs by Neural Embedding-based Semantic Similarity for Pattern Matching

Max Reinhard

Matrikelnummer: 359417

April 4, 2024

Supervised by Prof. Dr. Manfred Hauswirth
Additional guidance by Prof. Dr. Camille Roth*
and Dipl.-Math. Thilo Ernst†

*Centre Marc Bloch (An-Institut der Humboldt-Universität zu Berlin)

†Fraunhofer-Institut für offene Kommunikationssysteme

Abstract Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Contents

1. Introduction	5
1.1. References from the Future	5
1.2. Expose intro	6
2. Background	7
2.1. Semantic Hypergraph Framework	7
2.1.1. Structure and Syntax	7
2.1.2. Translation Process	9
2.1.3. Pattern Matching	10
2.2. Semantic Similarity Measures	12
2.2.1. Knowledge-based Semantic Similarity	13
2.2.2. Corpus-based Semantic Similarity	13
2.2.3. Neural Embedding-based Semantic Similarity	15
3. Problem Statement	19
3.1. Research Questions	21
3.1.1. Primary Question	21
3.1.2. Secondary Questions	21
4. Solution Approach	22
4.1. General Concept of NESS-SHMP	22
4.1.1. FNESS-SHPM	23
4.1.2. CNESS-SHPM	23
4.2. Implementing NESS-SHMP within the Semantic Hypergraph Framework . .	23
4.2.1. Extensions of the Pattern Language	23
4.2.2. Modification of the Pattern Language Processing	25
4.2.3. NESS Measurement Computations	25
5. Implementation	27
5.1. Relevant external Software Libraries used	27
5.2. Modules newly added to the SH Framework	27
5.3. Modifications of the SH Pattern Matching	27
5.3.1. Token position passing	27
5.3.2. SH PL implementation differences	27
5.4. Modifications to the Hypergraph database	27
6. Evaluation	29
6.1. Case Study: Conflicts	29
6.1.1. Expressions of Conflict	29
6.1.2. Reddit Worldnews Corpus	29
6.1.3. Semantic Hypergraph Patterns	30
6.2. Conflict Dataset	32
6.2.1. Base Edge Set	32

6.2.2.	Desired Characteristics	33
6.2.3.	Construction Process	33
6.2.4.	Edge Set Comparison	35
6.3.	Evaluation Process	35
6.3.1.	Evaluation Run Configurations	35
6.3.2.	Evaluation Metrics	37
6.4.	Evaluation Results	37
6.4.1.	Best F1-Score based Evaluation Run Comparison	39
6.4.2.	Evaluation Metric vs. Similarity Threshold	40
6.4.3.	Best F1-Score vs. Number of Reference Edges	42
6.4.4.	Predicate Lemma based Evaluation Run Comparison	42
6.5.	Result Discussion	47
6.5.1.	Retrieval Performance Improvement	47
6.5.2.	Retrieval Precision Behaviour	48
6.5.3.	Contextual Differentiation Ability	48
7.	Related Work	49
8.	Future Work	50
8.1.	Conceptual Improvements	50
8.2.	Implementation Improvements	50
8.3.	Further Evaluations	50
9.	Conclusion	51
A.	Appendix	59
A.1.	Reference Edge Sets	60
A.2.	Best F1-score based Eval. Run Comparison Tables	65
A.3.	Evaluation Metric Scores vs. Similarity Threshold Plots	67

1. Introduction

- Context: The big problem
- Problem statement: The small problem
- Methodology / Strategy
- Structure

Notes:

- Huge amounts of text, which can provide insight about stuff
- Automatic tools can provide assistance for humans to process all the text
- This generally means filtering the original text corpus or otherwise reducing amount of information the information that has to be processed by humans
- Filtering introduces a bias
- Especially for scientific purposes it is relevant to mitigate bias or at least understand what bias has been introduced (to make it transparent)
- Semantic Hypergraphs can be a valuable tool for that because...

Human life in times of widespread use of the internet and smartphones is most certainly more than ever interspersed with text-based communication...

Great progress has been made in the made in NLP, IR and IE in the past decade. This advancement of the state-of-the-art can primarily be attributed *Deep Learning* based methods, often also referred to as *Neural Networks*. (Hirschberg and Manning 2015; Min et al. 2023; Young et al. 2018)

1.1. References from the Future

open-opaque / strict-adaptive categorisation dimensions

1.2. Expose intro

A significant part of the social world is nowadays being represented by digitally manifested text. Examples for this range from instant messaging, social media and any form of collective web activity to encyclopaedic websites, digitized libraries and government intelligence. The amount and richness of available social text makes it a valuable data source for social science research while simultaneously creating an interest in automatic systems to analyze these texts on a large scale (Evans and Aceves 2016). Such research can be understood as part of the domain of *Computational Social Science* (CSS) (Lazer et al. 2009).

Systems based on techniques from the field of *Natural Language Processing* (NLP), as well as the interlinked fields of *Information Retrieval* (IR) and *Information Extraction* (IE), have demonstrated great success in a variety of task related to text analysis. This success is largely attributed to the advancements of applying of *Machine Learning* (ML) and especially *Deep Learning* (DL) methods to text (Hirschberg and Manning 2015) (Qiu et al. 2020). While being very effective at predicting or decision making, ML- and specifically DL-based systems generally do not deliver an explanation for their judgement and can mostly be viewed as "black box models" that are not transparent in their prediction or decision making process (Rudin 2019). Conversely this transparency and explainability is of high interest in CSS applications such as predicting political opinion based on social media activity (Wilkerson and Casas 2017).

The *Semantic Hypergraph* (SH) (Menezes and Roth 2021) is a framework for representing and analyzing *Natural Language* (NL). NL sentences can be modelled as an ordered, recursive hypergraph which can be represented in a formal language. The framework allows to specify semantic patterns in this formal language which can be matched against an existing SH. It aims to provide an *open* and *adaptive* system to analyse text corpora, especially in the domain of CSS. The framework is *open* in the sense that its representation formalism is inspectable and intelligible by humans and that the pattern matching follows explicit rules. The framework is *adaptive* in the sense that the parsing is built from adaptive, ML-based subsystems and therefore allows for an error-tolerant parsing from *NL* to *SH* in regards to grammatical and syntactical correctness.

2. Background

In this chapter the necessary background for this work is presented. This includes on the one hand the Semantic Hypergraph (SH) framework and its Pattern Matching capabilities in particular. On the other hand the concept of Semantic Relatedness and specifically Semantic Similarity is explained with a focus on its realisation in the of form neural embedding based semantic similarity (NESS).

2.1. Semantic Hypergraph Framework

The Semantic Hypergraph framework, introduced in Menezes and Roth 2021, offers a novel approach for the representation and analysis of natural language (NL) text. At its core, the SH framework transforms text into a structured, symbolic format, thereby facilitating a sophisticated analysis of text collections. A pattern language intrinsic to the SH framework enables the definition of patterns that encapsulate generalisations of text in the SH format.

Designed primarily for computational social science (CSS) researchers, the SH framework emphasises a high degree of openness, encompassing transparency and explainability. Additionally, the SH framework exhibits a level of adaptiveness, attributing to its reliance on Machine Learning (ML) components for the translation of NL text into the SH format. This adaptiveness reflects the framework’s capability to evolve and accommodate the variable form of natural language.

One of the objectives of the SH framework is to bridge the existing gap in open-adaptive natural language processing (NLP) systems. By providing a mechanism for structured symbolic representation and pattern-based analysis of text, the SH framework proposes a solution that combines the benefits of machine learning adaptiveness with the requisites of open and explainable research in CSS and beyond.

For a more comprehensive description of the Semantic Hypergraph framework, the original publication (Menezes and Roth 2021) should be consulted.

2.1.1. Structure and Syntax

In the SH framework, natural language is represented as a recursive, ordered *hypergraph* (in the following also simply referred to as *graph*). Such a hypergraph consists of *hyperedges* (in the following also referred to as *edges*). These hyperedges themselves can consist of a generally unrestricted number of hyperedges. To denote the hierarchy between hyperedges the terms *parent edge* and *child edge* or *sub edge* are introduced. If an edge e_A contains edges e_B and e_C , e_A is considered to be the parent edge of child or sub edges e_B and e_C . Since hyperedges are ordered, the edge $e_A = (e_B e_C)$ is different to an edge $(e_C e_B) \neq e_A$. If a hyperedge is not a composition of multiple hyperedges, it is called an *atom*.

Hyperedge Examples

Some example phrases and their corresponding hyperedges are given to help illustrate the general concept of the hypergraph representation and the hyperedge notation:

- "banana": **banana/C**, "eat": **eat/C**, "sweet": **sweet/C**
- "Mango is tasty.": (is/P.sc mango/C tasty/C)
- "Sophie throws the ball.": (throws/P.so sophie/C (the/M ball/C))

Hyperedge Components and Notation

An atomic hyperedge is noted as follows:

$$\begin{aligned} &\langle \text{CONTENT} \rangle / \langle \text{TYPE} \rangle \text{ or} \\ &\langle \text{CONTENT} \rangle / \langle \text{TYPE} \rangle . \langle \text{ARGUMENT ROLES} \rangle \end{aligned}$$

The different edge components *content*, *type* and potentially *argument roles* are here represented by placeholders in brackets (<>). In an atom, the content is always noted in lowercase. Non-atomic hyperedges are noted as a composition of their sub edges, as illustrated above. They also always have content and type, which are not part of the notation, but can be inferred. Each of the hyperedge components is explained in the following.

Hyperedge Types

Every hyperedge in a semantic hypergraph has a specific type. All types that an edge can assume are listed in table 2.1, which is adapted from Menezes and Roth 2021, p. 7.

Code	Type	Purpose	Example
C	Concept	Define atomic concepts	apple/C
P	Predicate	Build relations	(is/P berlin/C nice/C)
M	Modifier	Modify a concept, predicate, modifier, trigger	(red/M shoes/C)
B	Builder	Build concepts from concepts	(of/B capital/C germany/C)
T	Trigger	Build specifications	(in/T 1994/C)
J	Conjunction	Define sequences of concepts or relations	(and/J meat/C potatoes/C)
R	Relation	Express facts, statements, questions, orders, ...	(is/P berlin/C nice/C)
S	Specifier	Relation specification (e.g., condition, time, ...)	(in/T 1976/C)

Table 2.1.: Hyperedge types with their purpose and examples. For the examples the sub edge(s) that correspond(s) to the respective type is noted in bold face.

Type Inference Rules The type of a parent hyperedge is inferred based on the types and the order of its child hyperedges following specific inference rules. These rules can be inspected in Menezes and Roth 2021, p. 8, but are omitted here because they are deemed mostly irrelevant for this work.

Hyperedge Content

The textual representation of an edge is referred to as *edge content* or *edge text*. For atoms this is generally a word, except for special symbols (see below). Generally speaking, a hyperedges content is the reconstruction of the word, phrase or sentence that it represents, which can be produced by following the hyperedge structure.

Special Content Symbols The content of edges of the builder and conjunction type can also take on the form of special symbols $+/B$ and $:/J$. This is utilised to represent phrases where there is no (meaningful) word or character attributable to the edges content. Examples of this are (taken from Menezes and Roth 2021, p. 7):

- "guitar player": $(+/B \text{ guitar}/C \text{ player}/C)$
- "Freud, the famous psychiatrist": $(:/J \text{ freud}/C (\text{the}/M (\text{famous}/M \text{ psychiatrist}/C)))$

Argument Roles

To make the meaning of the relation of sub edges in a parent edge more explicit, builder and predicate edges can have argument roles. In parent edges that start with an edge of one of these two types, the roles of the other edges can be specified by adding argument roles to the builder or predicate edges. These are denoted as shown in section 2.1.1. A list of all possible argument roles can be inspected in (Menezes and Roth 2021, p. 8).

2.1.2. Translation Process

For natural language text to be represented in the Semantic Hypergraph framework, it needs to be translated (or parsed). This aspect of the framework is mostly irrelevant for this work in is therefore only described briefly here. The translation is realised by machine learning based software components. This is a high-level description of the process:

1. Segment text into sentences and tokenise sentences
2. Annotate tokens with POS-tags, dependency labels and NER categories
3. Use a trained classification model to predict an atom edge type for a token using its annotation labels as input features
4. Recursively construct more complex hyperedges from the sequence of atomic edges by essentially applying the type inference rules

The edges that correspond to the sentences into which the original input text has been segmented are referred to as *root edges* in the following.

add argument roles to make later pattern machings easier to understand?

2.1.3. Pattern Matching

Text represented in the SH framework can be analysed via *Semantic Hypergraph Pattern Matching* (SHPM). Information can be extracted from texts represented as hypergraph by constructing patterns in the frameworks pattern language and matching these patterns against the graph. The extracted information can take on the form of e.g. specific kinds of expressions and parts of these expression that fulfil a certain role.

Pattern Language

The *Semantic Hypergraph Pattern Language* (SHPL) essentially allows for generalisations of hyperedge structure and content. All valid hyperedges are generally also valid patterns in the SHPL. Table 2.2 list all of the pattern languages elements (also referred to as operators), which are relevant to this work. Their functionality and therefore the form of generalisation that they enable is also described.

Element	Notation	Function
Variable	VAR	Matches any edge (restrictable by type and argument roles). Returns the edge that it captures.
Wildcard	*	Matches any edge (restrictable by type and argument roles).
Options list	[...]	Matches one of the given options. Applicable to edge type and content.
Innermost atom	>	Matches against the innermost atom of the edge that it captures. Removes an arbitrary level of nested modifiers.
Lemma function	lemma	Matches if the lemma of the captured edge content equals the given word. Combinable with an options list. Applicable only to atoms.
Argument role set	$\{\alpha\beta\gamma, \epsilon\}$	Allows for argument roles in arbitrary order. Comma-separated argument roles are optional

Table 2.2.: Pattern language elements with their notation and function

Matching Process

At its core, the matching process involves comparing a predefined pattern against either an entire hypergraph, a set of edges, or a single edge. Independent of the matching target, this comparison is conducted one edge at a time, determining whether this edge aligns with the pattern. If pattern and edge do not match the result of this process is only this binary information of *No Match*. If pattern and edge do match, the result of the matching process depends on whether the pattern contains variables. In this case, the result is not only the binary information of *Match*, but also the assignment of the variables contained in the pattern. It is possible for a single edge to yield multiple variable assignments, however this aspect will not be discussed further as variable assignments hold limited relevance for this work.

The matching process works recursively, processing both the structure and content of the edge in relation to the pattern. A detailed discussion of this process is delegated to chapter 5. If a generalising part of the pattern (sub pattern) structurally matches a sub edge of the candidate edge, this sub edge is considered to be *captured* by this sub pattern. This is relevant for returning variable assignments and further processing as is the case with the innermost atom operator or any content generalisation functionality.

maybe
more spe-
cific ref

Assuming the matching the marching process is examining a specific sub pattern and sub edge, the content defined in the sub pattern is referred to as the (*pattern*) *reference content*. It is worth noting that a pattern may exclusively consist of structural elements, without specifying content. The content of the sub edge under comparison is termed as the (*edge*) *candidate content*. The process of matching involves a comparison between the reference content and the candidate content, employing either a direct string-based comparison or leveraging a form of content generalisation. Specifically this could be comparing the candidate word against a list of reference words or comparing the candidate words lemma against the reference word (or a list of reference words).

Edge argument roles are ignored when matching, if they are not specified in a pattern. The matching of argument roles is not elaborated further, as it is of little relevance here.

Example Matchings To illustrate the expressiveness and of the SHPL and the functionalities of the different language elements, some example matchings of patterns against edges are shown in table 2.3. The corresponding edges and patterns are listed below.

Example Edges

$e_1 = (\text{plays/P.so ann/C (acoustic/M guitar/C) })$
 $e_2 = (\text{plays/P.so ann/C (acoustic/M piano/C) })$
 $e_3 = (\text{plays/P.so ann/C (aggressive/M rugby/C) })$
 $e_4 = (\text{plays/P.sx ann/C (if/T (has/P.so she/C time/C) }))$
 $e_5 = (\text{play/P.so (and/J ann/C bob/C) (a/M song/C) })$
 $e_6 = ((\text{again/M perform/P.so }) \text{ they/C (the/M (same/M song/C) }))$

Example Patterns

$p_1 = (\text{plays/P PERS/C THING/C })$
 $p_2 = (\text{plays/P ann/C * })$
 $p_3 = (\text{plays/P ann/C VAR/[CS] })$
 $p_4 = (\text{plays/P ann/C >[guitar, piano]/C })$
 $p_5 = (\text{plays/P ann/C >THING/C })$
 $p_6 = (\text{plays/P ann/C (>INST/C [guitar, piano] }))$
 $p_7 = ((\text{lemma play/P }) \text{ ACTOR/C THING/C })$
 $p_8 = ((\text{lemma >[play, perform]/P }) \text{ ACTOR/C THING/C })$

Pattern	Edge	Match?	Variables
p_1	e_1	Match	PERS=ann/C, THING=(acoustic/M guitar/C)
p_1	e_4	No Match	
p_2	e_1, e_2, e_3, e_4	Match	
p_2	e_5, e_6	No Match	
p_3	e_1	Match	VAR=(acoustic/M guitar/C)
p_3	e_4	Match	VAR=if/T (has/P.so she/C time/C)
p_4	e_1, e_2	Match	
p_4	e_3, e_4	No Match	
p_5	e_1	Match	THING=guitar/C
p_5	e_3	Match	THING=rugby/C
p_5	e_4	No Match	
p_6	e_2	Match	INST=piano/C
p_6	e_3, e_4	No Match	
p_7	e_1	Match	ACTOR=ann/C, THING=(acoustic/C guitar/C)
p_7	e_5	Match	ACTOR=(and/J ann/C bob/C), THING=(the/M song/C)
p_7	e_6	No Match	
p_8	e_6	Match	ACTOR=they/C, THING=(the/M (same/M song/C)

Table 2.3.: Example matchings of patterns against edges

2.2. Semantic Similarity Measures

Determining the *Semantic Relatedness* (SR) *Semantic Similarity* (SS) of two text items is a central task in the field of natural language processing, information retrieval and information extraction. A text item can take on the form of a word, phrase, sentence paragraph or entire document. While SR and SS are often used synonymously, semantic relatedness can also be understood to entail a more nuanced description of the semantic relations of text items (Chandrasekaran and Mago 2021; Harispe et al. 2015). In the context of this work semantic similarity is therefore used to refer to the task of measuring the closeness, distance or similarity in regard to the meanings conveyed by two text items. This has to be differentiated from lexical (i.e. string-based) similarity measures, which do not rely on the meaning of text items (P. and Shaji 2019).

There exists a great variety of approaches to measuring semantic similarity and their categorisation also takes on different forms in the literature. A structure that is in line with most of the reviewed literature employs the following categories: *Knowledge-based Semantic Similarity* (KBSS), *Corpus-based Semantic Similarity* (CBSS) and hybrid approaches which combine techniques from the former two categories (Chandrasekaran and Mago 2021; Han et al. 2021; Harispe et al. 2015; Zad et al. 2021).

Depending on the exact definition of the categories, NNSS can also be seen as a sub category of CBSS. It will be treated as an individual category here, as this differentiation is deemed valuable, since it is the most relevant type of semantic similarity measure for this work. More specifically, the focus lies on a specific type of neural network-based semantic similarity measures, which is referred to as *Neural Embedding-based Semantic Similarity* (NESS).

rewrite
this!

2.2.1. Knowledge-based Semantic Similarity

Knowledge-based semantic similarity measures rely on some form of underlying structured knowledge source to compute the similarity between two text items. These knowledge sources (referred to as knowledge systems in the following) may be ontologies, knowledge bases, knowledge graphs, semantic graphs, (lexical) databases and similar systems (Chandrasekaran and Mago 2021; Harispe et al. 2015). Examples of such structured knowledge systems are *ConceptNet* (Speer, Chin, and Havasi 2017), *WordNet* (Miller 1995), the *Paraphrase Database* (PPDB) (Ganitkevitch, Van Durme, and Callison-Burch 2013), *Wikidata* (Vrandečić and Krötzsch 2014), *DBpedia* (Auer et al. 2007) and *Yago* (Suchanek, Kasneci, and Weikum 2007).

Depending on the specific structure of the knowledge system, different methods of computing the semantic similarity are possible. Given two text items, they first need to be mapped with knowledge items, often called *concepts* in the system. Then the similarity of these concepts can be calculated. There exists a variety of methods to calculate the similarity of two concepts, which can be categorised as follows: *structural or path-based*, *feature-based* and *information content (IC)-based*. Hybrid methods which combine multiple approaches also exist (Chandrasekaran and Mago 2021; Harispe et al. 2015).

Since most of the aforementioned knowledge systems have a graph-like structure, one type of measures is based on the spatial structure of the graph. The calculation of the similarity of two items is therefore mostly based on the length(s) of the path(s) between them. Another type of measure utilises the features that the two concepts exhibit, where the specific set of features is dependent on the employed knowledge system. A comparison of the two items feature values is the basis for calculating their similarity. IC-based methods rely on the "information provided by the concept when appearing in a context" (Sánchez and Batet 2013). This is closely related to the specificity of words measured by such metrics as *term frequency - inverse document frequency* (TF-IDF) (Aizawa 2003). Some of these methods determine the information content of concepts by using only information intrinsic to the knowledge system (or multiple knowledge systems). Other methods (additionally) utilize external information such as text corpora, so that the similarity measure could also be categorised as corpus-based. For a more in-depth exploration of knowledge-based similarity measures confer Chandrasekaran and Mago 2021; Harispe et al. 2015; Mihalcea, Corley, and Strapparava 2006; Zhu and Iglesias 2017.

2.2.2. Corpus-based Semantic Similarity

Corpus-based semantic similarity measures generally derive the semantics of text items and therefore their similarity not from a structured knowledge source, but from corpora of unstructured text. An assumption that is central to most of the approaches in this category is the *Distributional Hypothesis*, which states that words occurring in the same contexts tend to have similar meanings (Harris 1954). Consequently these approaches are also referred to as *Distributional Semantic Similarity Measures* (Mohammad and Hirst 2012b). Although not all corpus-based semantic similarity measures are based on the distributional hypothesis, it is considered to be the dominant and most relevant type of approaches in this category (Harispe et al. 2015, Section 2.4).

Vector Space Model

The Vector Space Model (VSM) (Turney and Pantel 2010) is the underlying framework for the text representation used by distributional semantic similarity measures. In this model, vectors represent text items, typically words, and are designed to encapsulate the distribution of words within a corpus. This generally involves counting word occurrences within particular contexts and organising these counts into a matrix. If the context is defined as a document (assuming the corpus is structured into distinct documents), the matrix is known as a *term-document* or *word-document frequency matrix*. Here, rows represent words and columns signify documents, with each cell indicating the frequency of a word's occurrence in a document. The context may also be defined more broadly as e.g. paragraphs, sentences, phrases, or a text window of arbitrary size (around another word). The entries in the resulting *word-context frequency matrix* can be weighted to reflect their specificity to the context, using measures such as TF-IDF or *(Positive) Point-wise Mutual Information* (Church and Hanks 1989; Niwa and Nitta 1994). Due to the potentially large vocabulary size and high number of context items, the word-context frequency matrix – and consequently the derived word vectors – may be high dimensional and sparse. It has been demonstrated that semantic similarity measurements benefit from techniques that smooth the matrix and therefore reduce the dimensionality of the vectors (Deerwester et al. 1990). A commonly employed method for this purpose is *Singular Value Decomposition* (SVD), or specifically its variant *truncated SVD*. Though other approaches, such as *Nonnegative Matrix Factorisation* (Lee and Seung 1999) or filtering of low entropy dimensions (Lund and Burgess 1996) have also been explored. A principle limitation of these approaches is, that they treat the context as a *bag of words* – ignoring word order beyond word proximity.

Vector Similarity Measures

In assessing the similarity between two vectors within the VSM, a variety of measures can be utilised to quantify their relationship. Spatial similarity measures treat the VSM as a geometric construct and determine a spatial relation of the vectors. The most notable examples include the Manhattan distance (L1 norm), the Euclidean distance (L2 norm), and cosine similarity. Alternative measures are mostly motivated by an information theory or probabilistic perspective on the VSM. Relevant examples among these are the Kullback-Leibler divergence and the Jensen-Shannon divergence (Mohammad and Hirst 2012b,c). It shall be noted, that a probabilistic interpretation entails constraints for the characteristics of the VSM. The choice of a suitable vector similarity measure is generally dependent upon the specific construction of the vectors in use. Cosine similarity, in particular, has been highlighted for its effectiveness (Mohammad and Hirst 2012a; Turney and Pantel 2010).

Spatial VSM-based Semantic Similarity Measures

This work focusses on spatially interpreted VSMs and their associated similarity measures, since they can be seen as a precursor to the neural embedding-based similarity measures, which are outlined and further elaborated on in section 2.2.3. In the following some distributional semantic similarity measures are presented, which were relevant for the development of the field. For each approach, the definition of the context for the word frequency matrix and the matrix' weighting and smoothing (i.e. dimensionality reduction) procedure are noted.

Latent Semantic Analysis (LSA) (Deerwester et al. 1990; Landauer and Dumais 1997; Landauer, Foltz, and Laham 1998): In LSA, the context for the frequency matrix is a paragraphs. The matrix undergoes dimensionality reduction through a variant of Singular Value Decomposition (SVD), which reduces columns but retains rows, aiming to preserve the similarity structure among words. The similarity between word vectors, represented by their row values, is calculated using cosine similarity.

Hyperspace Analogue to Language (HAL) (Lund and Burgess 1996): HAL employs a sliding window to define its context, creating a word co-occurrence matrix with association strength values that depend on word proximity. Dimensionality reduction is achieved by excluding columns with low entropy. The vector similarity measure used is either Euclidean or Manhattan distance.

Correlated Occurrence Analogue to Lexical Semantics (COALS) (Rohde, Gonnerman, and Plaut 2006): COALS is a improved version of HAL and also uses a sliding window for its context. The matrix undergoes correlation normalisation and possibly SVD for smoothing and dimensionality reduction. Cosine similarity is used as vector similarity measure.

Explicit Semantic Analysis (ESA) (Gabrilovich, Markovitch, et al. 2007): ESA’s frequency matrix context are Wikipedia articles, also called concepts. Weighting of the word-concept matrix is performed using TF-IDF. Semantic similarity is measured using cosine similarity between weighted concept vectors. The reliance on Wikipedia structure arguably renders ESA a hybrid combination of corpus- and knowledge-based approaches.

2.2.3. Neural Embedding-based Semantic Similarity

In natural language processing, the term *embedding* was first used in the context of Latent Semantic Analysis (Deerwester et al. 1990) to describe a vector representation of text. Since then, *text embedding* has become widely used in NLP to describe a dense, fixed-length vector representations of text items, which can be interpreted spatially to measure semantic relations between them (Almeida and Xexéo 2023).

Strictly speaking, *neural embeddings* of text refer to representations that have been produced by a neural network model. The development of these approaches is deeply interlinked with the advent of *Neural (Network) Language Modelling* (NNLM). The task of *language modelling* traditionally is to predict the most probable next word given a sequence of previous words (Chen and Goodman 1999). While Bengio, Ducharme, and Vincent 2000 are commonly referred to as the first to use a neural network for language modelling, Collobert and Weston 2008 are seen to be the first to build a NNLM with the focus on producing embeddings.

Neural Embedding-based Semantic Similarity Measures (NESS) compute the semantic similarity of text items based on a spatial measurement between their respective neural embeddings. NESS measures can be considered as a subform of CBSS, since they also fundamentally rely on the distributional hypothesis and employ the same general approach of deriving distributional semantics from a text corpus. Here, neural embedding-based approaches are further differentiated into *Fixed Word Embedding-based Semantic Similarity Measures* (FNESS) and *Contextual Embedding-based Semantic Similarity Measures* (CNESS), which is elaborated in the following.

Fixed Word Embeddings vs. Contextual Embeddings

In this work, (*fixed*) *word embeddings* refer to neural text embeddings produced by models that are limited to single words. For a or given word, such a model will always generate the same embedding, hence the specification *fixed* (or *static*) word embedding. A significant consequence of this limitation is, that these word embedding models are not able to differentiate *homonyms*¹ or more specifically *homographs*² (Li and Yang 2018). These differentiations are only possible by taking the context in which the word appears into consideration. The context may be a phrase, sentence, paragraph or any sequence of tokens. A token can be word, but also the result of another kind of segmentation of the input text. While a sequence of words is generally processable using a fixed word embedding model, this results in a sequence of independently generated word embeddings. In the case of *contextual embedding* models, the generation of a token embedding depends on the other tokens in the sequence. This may result in different token embeddings for the same token, if it appears in different contexts. Therefore contextual embeddings are principally able to represent different meanings of homographs and may also differentiate more subtle variations of word meaning, that can be derived from context (Liu, Kusner, and Blunsom 2020).

Deep Learning-based Semantic Similarity

Deep Learning-based Semantic Similarity Measures (DLSS) are often treated as a distinct category in the literature, which this is not necessarily true for non-deep machine learning based approaches (Chandrasekaran and Mago 2021; Han et al. 2021; Harispe et al. 2015; Zad et al. 2021). In this work, we do not treat DLSS as a separate category, instead it is subsumed by the category of NESS. Deep learning-based approaches employ deep neural network models, which have multiple hidden layers, allowing them in principle to learn hierarchical of representations of the input data (Goodfellow, Bengio, and Courville 2016). Given a text item as input, the state of a specific hidden layer in the trained network after an inference pass is generally considered to be the embedding of the text item. From this perspective, all DLSS measures can be seen as neural embedding-based. However, there also exists a variety of deep learning-based approaches, which use models that directly predict the semantic similarity between two input text items in the output layer. These methods are not in the focus of this work, for such approaches confer e.g. He and Lin 2016; Lopez-Gazpio et al. 2019; Shao 2017; Tai, Socher, and Manning 2015; Tian et al. 2017; Tien et al. 2019a,b; Wang, Mi, and Ittycheriah 2017; Zheng et al. 2019.

should
this be
moved?

Fixed Word Embedding-based Semantic Similarity

Fixed Word Embedding-based Semantic Similarity Measures (FNESS) in this work denotes semantic similarity measures that employ fixed word embedding models. Here, the concept of *neural embeddings* extends to encompass not only those embedding techniques directly derived from neural models, but also includes approaches that incorporate elements of neural model methodologies to achieve outcomes that rival strictly neural embeddings.

¹*homonym*: "A word that both sounds and is spelled the same as another word"
(<https://en.wiktionary.org/wiki/homonym>)

²*homograph*: "A word that is spelled the same as another word [...]"
(<https://en.wiktionary.org/wiki/homograph>)

This broader interpretation aligns with perspectives found in existing literature (Sezerer and Tekir 2021; Zuccon et al. 2015). Fixed word embedding models are generally based on non-deep machine learning methods and can be further divided into *prediction-based*, *count-based* and *hybrid* or *meta* methods. In the following, some of the most relevant word embeddings models are presented.

Prediction-based Methods These methods are based on the concept of language modelling and work with local context windows. They are trained on predicting a target word (or token) given a context window of words (or tokens) or the other way around.

Word2Vec (Mikolov, Yih, and Zweig 2013; Mikolov et al. 2013a,b): Word2Vec has two forms; the *Continuous Bag of Words* (CBOW) model and the skip *Skip-Gram* (SG) model. CBOW predicts the target word given a context and SG predicts each context word given a target word. Both models are log-(bi)linear models, which basically learn a logistic regression with one hidden layer, whose weights are then used as embeddings. The best performing variant of Word2Vec is *Skip-Gram with Negative Sampling* (SGNS), which produced state-of-the-art results at the time.

fasttext (Bojanowski et al. 2017): fasttext is a better performing extension of Word2Vec. The central modification is, that the model works on subword tokens, which helps it to generalise to word which are unknown from the training data.

Count-based Methods These methods are no neural embedding models in a strict sense, since they do not employ a neural network architecture. In terms of approach, they fall in line with spatially interpreted VSMS based on constructing a word-context frequency matrix with word counts (Almeida and Xexéo 2023; Turney and Pantel 2010), which are discussed in section 2.2.2. In distinction to these approaches, the methods presented here also incorporate local context window information inspired by NNLM approaches and perform competitively when compared with neural embeddings.

GloVe (Pennington, Socher, and Manning 2014) GloVe leverages corpus-wide, hence *global*, word co-occurrences represented as word-word frequency matrix populated with probability ratios. The word co-occurrence probability ratios are weighted based on the distance of the word pairs in the context window from which they are extracted. It outperforms SGNS, given the same training corpus, training time and context window size.

LexVec (Salle, Idiart, and Villavicencio 2016; Salle and Villavicencio 2018, 2019; Salle, Villavicencio, and Idiart 2016): LexVec is based on the low-rank, weighted factorisation of the positive point-wise mutual information matrix via stochastic gradient descent. It incorporates local context window information by applying a weighting scheme to the matrix that has been shown to be implicitly performed by SGNS (Levy and Goldberg 2014). There also exists a subword based variant of LexVec, which shows competitive performance compared to fasttext.

Hybrid and Meta Methods *Hybrid word embedding* methods in this work refer to approaches which are not only trained on unstructured text corpora, but also incorporate some form of structured knowledge source. This is akin to the ESA method presented in section 2.2.2. *Meta word embedding* methods combine the information of multiple existing

word embeddings into a new embedding model (Bollegala and O'Neill 2022). Both approaches are applicable simultaneously, resulting in a *hybrid meta word embedding* model.

Conceptnet Numberbatch (Speer, Chin, and Havasi 2017, 2018): Conceptnet Numberbatch is hybrid ensemble model that integrates structured knowledge from ConceptNet with existing word embedding models. First, ConceptNet-specific word embeddings are constructed by representing the knowledge graph structure as a matrix of term-term connections, with cell values reflecting the sum of connecting edge weights. Further processing (i.a. smoothing and dimensionality reduction) is applied to this matrix and the resulting word embeddings are merged with Word2Vec, GloVe and fasttext embeddings. The method produced state-of-the-art-result at the time, outperforming LexVec and all subsumed embedding models.

MetaVec (García-Ferrero, Aggeri, and Rigau 2021): MetaVec is in principle a pure meta word embedding model which integrates fasttext, ConceptNet Numberbatch, *JOINTChyb* (Goikoetxea, Soroa, and Agirre 2018) and *Paragram* (Wieting et al. 2015). However, JOINTChyb leverages data from WordNet, Paragram utilises the PPDB and Conceptnet Numberbatch relies on ConceptNet as described above. Consequently MetaVec is highly influenced by structured knowledge as well and can therefore also considered to be a hybrid meta word embedding model. It outperforms ConceptNet Numberbatch on a number of evaluations and is seen to be the state-of-the-art at this time.

Contextual Embedding-based Semantic Similarity Measures

text is tokenized and token ids are used as input features

contextual neural embedding model

generally based on deep neural network models, as stated above

(SENTENCE-) BERT (SENTENCE-) T5 E5 (V2) GTE (E5-MISTRAL)

already
refer to
massive
text em-
bedding
bench-
mark at
this point?

3. Problem Statement

CSS researches may typically be interested in extracting statements of a specific kind from a text corpus, such as expressions of sentiment of an actor towards some entity or expressions of conflicts between different actors. Two sensible ways to frame this task are as *text classification* (Kowsari et al. 2019) or *text retrieval* (Manning, Raghavan, and Schütze 2008). It can be addressed with a wide range of system, which will be generally referred to as *automatic text analysis* systems in the following. These systems are mostly based on techniques from the field of Natural Language Processing (NLP), as well as the interlinked fields of Information Retrieval (IR) and Information Extraction (IE) (Chowdhary 2020).

A relevant perspective of categorising text analysis systems, especially from the point of view of CSS researchers, are the dimensions *open-opaque* and *adaptive-strict* (Menezes and Roth 2021). Here openness refers to the systems users ability to inspect and understand the processing, which we can also describe as transparency and explainability. These properties are of high interest in CSS applications such as predicting political opinion based on social media activity (Wilkerson and Casas 2017). An adaptive text analysis system does not (only) operate on strict rules, but is able to learn and modify its behaviour in some way. It is therefore in principle able to handle unforeseen variations in the text it processes. While both of these two properties are desirable for users are often found to be a trade-off. Current successful adaptive systems are most often based on neural networks (Hirschberg and Manning 2015), which are opaque in the way how they represent and process text (Rudin 2019).

The Semantic Hypergraph (SH) framework aims to fulfil both the open as well as the adaptive property of a text analysis system. It offers an inspectable and understandable representation of text that is constructed by a parser based on machine learning components. The SH representation and its construction can be therefore considered to fulfil the open-adaptive properties. The SH pattern matching language can be used to define patterns that match a specific subset of hyperedges in a given hypergraph. The matching process is purely symbolic and follows a set of fixed rules. It can therefore be considered to be open-strict. In the context of the SH framework the CSS research task described above is better framed as text retrieval. The SH pattern acts as a *query* for which the most relevant items are retrieved. While the SH frameworks capabilities are not restricted to text retrieval, the work is focused on this application.

The SH pattern defined by a user may among other things specify the structure of the edges that should match it as well as their type (and the types of possible sub-edges). The SH pattern language allows it to describe different levels of generalisations for the structural matching. Additionally the actual words that should match need to be specified i.e. the edge content to match against, if the edge matches the pattern structurally. These words need to be given explicitly and the only way of generalising is via the lemma functional pattern. This lack of generalisation capability entails that a bias is introduced into the

matching process by the manual selection of words by the SH frameworks user, who defines the pattern.

To better illustrate the problem hyperedge 2 and hyperedge 1 demonstrate how NL sentences are parsed to SH based on this simplified introduction the the SH representation.

(likes/P ann/C apples/C)

Hyperedge 1.: SH representation for the sentence "Ann likes apples"

(likes/P ann/C bananas/C)

Hyperedge 2.: SH representation for the sentence "Ann likes bananas"

hyperedge 1 and hyperedge 2 both follow the same structure, but differ in the content of the last sub-edge. Both edges are hence matched by pattern 1, which does not specify content for this sub-edge. The SH pattern language also allows to define a pattern that matches both hyperedge 1 and hyperedge 2 via a list of words as in pattern 4. However is not possible define a pattern that matches based on some form of *Semantic Relatedness* (SR) or *Semantic Similarity* (SS) (Harispe et al. 2015) regarding content. Referring to the example above this means using the SH framework it is not directly possible to retrieve every sentences that declares that "Ann likes *some kind of fruit*" or that "Ann likes *fruits similar to apples*". This former would require to provide a comprehensive list of every fruit while the latter would require the user to specify all fruits he deems similar to apples.

(likes/P Ann/C */C)

Pattern 1.: "Ann likes something" pattern

(likes/P ann/C [apples, bananas]/C)

Pattern 2.: "Ann likes apples or bananas" pattern

Utilizing some form of SR/SS regarding to edge content in the SH matching process would allow users to define patterns, which describe generalisations of edge content. There exists a great variety of approaches for determining the SR/SS of text, which can generally be divided into *Corpus-based Measures* and *Knowledge-based measures* (Harispe et al. 2015, Section 1.3.2). The latter approaches may generally provide the openness in the measurement determination that is desired by CSS researchers. However among the former recent ML-based and especially DL-based approaches have been outperforming most other approaches (Chandrasekaran and Mago 2021). They generally rely on computing a vector space representation (or embedding) of texts which can then be used to calculate their similarity and will therefore be referred to as neural embedding-based semantic similarity (NESS) measures.

Word semantics generally depend on textual context and hence does the SS between words (Harispe et al. 2015, Section 2.2.3). Incorporating contextuality when extending the SH pattern matching process by SS therefore poses a central challenge. Context-dependent SS would allow to specify matching edge content beyond isolated word semantics, although this may not always be desirable or necessary as in the example above.

This has to be adapted based on chapter 2
-> add reference to FNESS/CNESS and derive relevancy of both for this work -> modify RQs
-> derive in more detail why embedding based SS is chosen
-> argue why embedding models (-> caching embeddings / add discussing about efficiency?)
-> refer to the similarity threshold specifically when talking about controlling?

Integrating NESS measures into the pattern matching process would allow for edge content generalisation and therefore would make the process more adaptive. As illustrated earlier, NESS measures principally do not provide the openness that is inherent to the pattern matching process of the SH framework. In the sense of the open-opaque / strict-adaptive classification described above this integration would mean a shift from openness to opaqueness and from strictness to adaptivity. To counteract the opaqueness introduced by an NESS integration into the SH pattern matching, allowing user control over generalisation levels can maintain some openness while still benefiting from increased adaptivity.

3.1. Research Questions

Based on the problem statement outlined above, we pose the following research questions:

3.1.1. Primary Question

R Can neural embedding-based semantic similarity regarding edge content be integrated into the pattern matching of the Semantic Hypergraph framework to allow for more generalising patterns while providing control over the generalisation and therefore maintaining some openness of the pattern matching process?

3.1.2. Secondary Questions

R.1 What neural embedding models would be the most suitable for accurately assessing semantic similarity within the Semantic Hypergraph pattern matching process while effectively addressing the challenges posed by contextuality?

R.2 How can neural embedding based semantic similarity effectively (and efficiently?) be integrated into the Semantic Hypergraph pattern matching?

R.3 Does integration neural embedding-based semantic similarity improve the retrieval performance of the Semantic Hypergraph framework and how does it impact recall and precision when matching a pattern against a set of known desired matching results?

R.4 How can the level of edge content related generalisation in the pattern matching process be effectively and transparently controlled?

4. Solution Approach

The proposed solution to the research questions posed in section 3.1 is to conceptualise and implement the integration of Neural Embedding based Semantic Similarity into the pattern matching of the Semantic Hypergraph framework, followed by a suitable evaluation of this integration. This chapter outlines the concept of the approach, while the implementation and evaluation are detailed in chapter 5 and chapter 6 respectively.

The integration strategy involves pinpointing the most opportune point within the SH framework’s pattern matching for the inclusion of NESS. This allows to identify which parts of the SH framework require modifications to accommodate NESS integration, as well as recognising any components that are currently missing. To facilitate this, the following section elaborates on the approach’s general concept, addressing its core challenges and design decisions.

The system that is conceived by integrating NESS into the Semantic Hypergraph pattern matching will be referred to as *Neural Embedding-based Semantic Similarity extended Semantic Hypergraph Pattern Matching* or **NESS-SHMP**

4.1. General Concept of NESS-SHMP

The pattern matching process is described in section 2.1.3, where the concepts of edge candidate content and pattern reference content are introduced. At a certain point in the process, the candidate content is matched against the reference content. In the original form of the SH pattern matching, this is limited to exact string-based or lemma-based matching. The SH framework’s functionality will be extended by neural embedding based semantic similarity measurement based matching regarding edge content.

The NESS measurement employs a neural network model to generate embeddings for the candidate and reference content, which in the following are referred to as candidate embedding and reference embeddings. The distance between these embeddings is calculated using a suitable metric and then compared to a given semantic similarity threshold s_t . If the measured semantic similarity exceeds this threshold, the contents of the pattern and the edge are considered to match. This process requires that the candidate content, reference content, similarity threshold (plus possibly additional other relevant parameters) are specified and passed to the component responsible for this matching step.

Moreover, the integration of NESS-based content matching into the SH pattern matching process will take two forms: *Fixed Neural Embedding-based Semantic Similarity extended Semantic Hypergraph Pattern Matching* (FNESS-SHMP) and *Contextual Embedding-based Semantic Similarity extended Semantic Hypergraph Pattern Matching* (CNESS-SHMP). The particularities of these two system variants are elaborated in the following.

4.1.1. FNESS-SHPM

In the case of FNESS, embeddings are generated based on single words. Therefore the candidate content takes on the form of a *candidate word* while the reference content may be one or multiple *reference words*. It is considered potentially useful to leverage the generalisation capabilities that already exist in the SH pattern matching in the form of the lemma function. Hence a lemma based variant of FNESS extended SH pattern matching (LFNESS-SHMP) is also conceptualised. In the case of LFNESS, the candidate content takes on the form of the candidate words lemma.

4.1.2. CNESS-SHPM

In the case of CNESS, embeddings are generated based on context which can be phrases or complete sentences. In the SH framework those are represented in the form of hyperedges and therefore the candidate and content takes on the form of a *candidate edge* and one or multiple *reference edges*.

4.2. Implementing NESS-SHMP within the Semantic Hypergraph Framework

The Semantic Hypergraph framework offers two primary avenues for enhancement: through the expansion of its pattern language or through the adaptation or refinement of the software interface utilised in the matching process. These approaches are not mutually exclusive and can be effectively combined.

Leveraging the core principles of the SH framework, the most logical and straightforward method to enrich its pattern matching capabilities is by extending its pattern language. This strategy facilitates the concurrent application of NESS-based content matching alongside string- or lemma-based matching within the same matching process.

Therefore, the incorporation of NESS-SHMP into the existing SH framework encompasses three principal components: the extension of the pattern language, the requisite modifications to the pattern language processing and the actual NESS measurement calculations.

Here is probably the most suitable point to add a system/approach schema

4.2.1. Extensions of the Pattern Language

For the inclusion of NESS-based content matching within the SH pattern language, the *SemSim* functional pattern will be introduced. This addition is targeted at the functional pattern segment of the language, which is already predisposed to support flexible enhancements for SH pattern matching. This approach is consistent with the existing mechanism for integrating lemma-based matching capabilities into the framework.

SemSim Functional Pattern

When constructing a SH pattern that should be matched against a hypergraph, the SemSim functional pattern can be utilised at those points in the pattern, where content should be matched by NESS. The added SemSim functional pattern (also referred to as SemSim pattern) has the following syntactical structure:

adapt this to include the usage of a variable

<SF> <RC>/<HT>.<AR> <ST>

It consists of different syntactical components, here represented by placeholders that are enclosed in brackets (<>). These components are explained in the following:

SemSim Function (SF) The semsim function can be one of the following: `semsim-fix`, `semsim-fix-lemma` and `semsim-ctx`. These functions correspond to FNESS-, LFNESS- and CNESS-SHPM.

Reference Content (RC) The reference content is used to specify the reference word(s) for the FNESS variant. The square bracket notation which is already part of the SH pattern language is leveraged to pass multiple words as a list. The reference edges that are needed for the contextual NESS variant is not given via the semsim functional pattern itself, but as a software interface argument to the matching process due to practical considerations (see section 4.2.1).

Hyperedge Type (HT) and Argument Roles (AR) yada yada see above

extend
when
chapter
2 is ready

Similarity Threshold (ST) The similarity threshold s_t can optionally be specified for a specific occurrence of the SemSim functional pattern, since it can also be given as a global parameter of the matching process (see section 5.4).

Example Usages

To illustrate the usage of the SemSim functional pattern, some examples are given:

(likes/P Ann/C (semsim-fix apples/C))

Pattern 3.: "Ann likes something similar to apples" pattern

(likes/P ann/C (semsim-fix [apples, bananas]/C))

Pattern 4.: "Ann likes similar to apples or bananas" pattern

(likes/P Ann/C (semsim-fix mangos/C 0.5))

Pattern 5.: "Ann likes something similar to mangos" pattern with $s_t = 0.5$

Limitations of the Pattern Language Extension

The utilisation of the fixed variant of NESS-SHMP is possible solely via the SemSim functional pattern. This means that all information that is required to apply FNESS-based content matching, specifically the reference content and the similarity threshold, can be given in a SH pattern. In contrast, it was found impractical to give the reference edges needed for the contextual NESS variant via a SH pattern. While the reference content argument could generally be used for that, this would result in very exhaustive patterns and impair their readability for humans. Therefore the conceptual design choice was made to pass the reference edges via the software interface of the SH pattern matching.

4.2.2. Modification of the Pattern Language Processing

This section outlines the integration of SemSim functional pattern processing with the current pattern language processing. The process is triggered when a pattern, incorporating a SemSim pattern, is matched against a specific hyperedge and the SemSim pattern is reached without any prior mismatches. At this point, the SemSim pattern captures a sub-edge for processing, which includes the symbolic matching of the hyperedge type and argument roles, as well as extracting the necessary information for NESS measurement computations.

The operation of SemSim pattern processing changes based on the specific SemSim function applied. With FNESS-SHMP or LFNESS-SHMP, the process extracts the candidate word(s) or their lemma(s) from the identified sub-edge and the reference word(s) from the pattern. This extraction process requires the sub-edge to be atomic, which implies combination with the innermost atom operator to ensure functionality in all cases. For CNESS, it conducts a standard string-based matching of the reference content. Therefore utilizing the wildcard operator as a reference content argument is sensible in most practical use cases. For any NESS-SHMP variant, the similarity threshold is extracted from the SemSim pattern, if given

Existing components manage the symbolic matching of hyperedge type and argument roles, consistent with the standard pattern matching. The SemSim function, along with the similarity threshold (when provided), candidate words and reference words for (L)FNESS-SHMP, are passed to the newly designed NESS computation components.

4.2.3. NESS Measurement Computations

Upon reaching this phase, the NESS-SHPM process has already either extracted candidate and reference contents via pattern language processing (in FNESS-SHPM) or received them through software interface arguments (in CNESS-SHPM). Similarly, the similarity threshold has been either derived from the SemSim pattern or assigned as a default parameter through the software interface. Consequently, the process is set to obtain the respective embeddings for the candidate and reference content, which are crucial for conducting the similarity assessment.

How it
the NESS
model
specified?
talk about
ness con-
fig?

Fixed Neural Embeddings

The creation of fixed neural embeddings necessitates only the candidate and reference words. Given that each word maps directly to a specific embedding, a straightforward lookup in the chosen fixed neural embedding model suffices to produce the necessary embeddings for both candidate and reference.

Contextual Neural Embeddings

The development of contextual embeddings for the candidate and reference involves a sequence of steps, premised on the assumption that these embeddings should mirror the specific sub-edge of the candidate edge captured by the SemSim pattern and its analogous sub-edge(s) in the reference edge(s). Thereby the reference edges need to structurally match (with disregard to edge content) the pattern which contains the SemSim functional pattern, so that these reference sub-edges can be identified. To allow for verification of this assumption an alternative approach to constructing contextual neural embeddings does not account for sub-edge matching and only consists of the initial two stages of the construction process outlined below:

1. Reconstructing phrases or sentences from the candidate and reference edges (deemed context items).
2. Producing embeddings for these context items using the designated contextual neural embedding model.
3. Identifying the sub-edge within the reference that aligns with the candidate's sub-edge captured by the SemSim pattern.
4. Associating the identified candidate and reference sub-edges with their respective sub-embeddings derived from the context item embeddings.
5. Generating the final embeddings for both candidate and reference based on these sub-embeddings.

Embedding Similarity Measurement

The procedure calculates the cosine similarity between the embeddings of candidate and reference content, comparing it to the pre-established similarity threshold. If the resultant similarity score surpasses this threshold, the candidate and reference contents are considered to match. Given multiple reference content items (and therefore multiple reference embeddings), the pairwise similarities between the candidate and the reference embedding is computed. The maximum of these similarities is then compared to the similarity threshold to assess whether the candidate and reference contents match.

why cosine similarity?
why max similarity?

5. Implementation

5.1. Relevant external Software Libraries used

Here list libs and models to be referenced later.

Word2Vec via Gensim SentenceTransformers Transformers SpaCy

5.2. Modules newly added to the SH Framework

Semsim instances

reference edge sample modification parameter

5.3. Modifications of the SH Pattern Matching

skip semsim

root edges

5.3.1. Token position passing

The matching process of the SH framework is extended by passing the tok pos down (this references implicitly the recursive implementation of the pattern matching which might be too technical)

5.3.2. SH PL implementation differences

Differences between the formal notation and the notation used in the implementation → innermost atoms operator, variable functional pattern etc. does this need to be included?

5.4. Modifications to the Hypergraph database

is this really necessary? tok pos etc, but not actually specific to semsim

NESS configuration Additional parameters need to be given to the NESS-SHMP process, which includes the specific neural embedding model to use

- Default similarity threshold
- Specific NESS model
- CNESS embedding prefix
- CNESS all tokens option

6. Evaluation

In this chapter the conceived concept (see chapter 4) and specific implementation (see chapter 5) of the NESS-SHPM system is being evaluated to answer the research question(s) posed in section 3.1. Therefore a case study is conducted to evaluate the system for a specific use case.

refer to the RQs more specifically? how are they going to be answered?

6.1. Case Study: Conflicts

The conflicts case study follows the approach presented in Menezes and Roth 2021, where expressions of conflict are extracted from a given SH using a single SH pattern. In their work they build upon the information extracted by the pattern to conduct further analyses, which are not in the scope of this work. Here the evaluation is limited to the task of classifying whether the content of a given edge in the SH is an expression of conflict or not. Or framed differently, the task is to retrieve exactly all those edges whose content is an expression of conflict. The evaluation will compare the retrieval performance of a suitable set of different SH patterns and corresponding configuration of the NESS-SHMP system by matching them against a labelled dataset of hyperedges.

should I explain why specifically the conflicts and not some other case study (i.e. dataset) -> because there was none... but then I need to show why there was none and what are the criteria for a case study to be suitable to evaluate the system

6.1.1. Expressions of Conflict

An expression of conflict in the context of this case study is defined as a sentence which fulfils the following properties:

There is a conflict between two explicitly named actors, wherever these actors are mentioned in the sentence; whereby a conflict is defined as antagonizing desired outcomes.

6.1.2. Reddit Worldnews Corpus

The corpus from which those expressions of conflict are retrieved consists of news titles that were shared on the social media platform *Reddit*. Specifically all titles shared between January 1st, 2013 and August 1st, 2017 on *r/worldnews*, which is described as: “A place

for major news from around the world, excluding US-internal news.”¹ This corpus contains 479,384 news headers and is in the following referred to as the *Worldnews-Corpus*.

Each of these headers is comprised of a single sentence and is forms a root edge in the SH constructed from it. In the following this SH is referred to as the *Worldnews-SH*. Parsing errors that may potentially occur during this constructed and can obstruct a correct retrieval of a wrongly parsed edge i.e. wrongly represented sentence. These errors are out of scope of this work. All edges in the Worldnews-SH are assumed to be correctly parsed.

6.1.3. Semantic Hypergraph Patterns

The SH patterns that are used in this evaluation all have the same general form to isolate the effect of replacing a purely symbolic matching against a specific word or list of words with NESS-SHMP. In this section the general form of these pattern will be described, which entails consequences for the creation of the labelled dataset described in section 6.2.

Original Conflict Pattern

Pattern 6 is originally defined in Menezes and Roth 2021, p. 22 and is therefore referred to as the *original conflict pattern*. It is used to extract conflicts between two parties **SOURCE** and **TARGET**, potentially regarding some **TOPIC**. As mentioned before, the assignment of these variables is irrelevant for this case study.

The original conflict patterns contains two sub-patterns which utilize word lists. These sub-patterns match the trigger sub-edge and predicate sub-edge of a candidate edge respectively and are in following referred to as *trigger sub-pattern* and *predicate sub-pattern*. If not stated otherwise these terms will refer to pattern 6.

- **Trigger sub-pattern:** [against,for,of,over]/T
- **Predicate sub-pattern:** (PRED/P.so,x) \wedge
(lemma/J >PRED/P [accuse,arrest,clash,condemn,kill,slam,warn]/P)

In the trigger sub-pattern the content of the candidate trigger sub-edge is directly matched against a list of prepositions, which are in the following referred to as the *conflict prepositions*. In case of the predicate sub-pattern, the word list is matched against the lemma of the innermost atom of the candidate predicate sub-edge, which is always a verb. The list of verbs used here will in the following be referred to as the *conflict verbs*.

- **Conflict prepositions:** against, for, of, over
- **Conflict verbs:** accuse, arrest, clash, condemn, kill, slam, warn

$$(\text{PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]}) \wedge \\ (\text{lemma/J >PRED/P [accuse,arrest,clash,condemn,kill,slam,warn]/P})$$

Pattern 6.: Original conflict pattern

¹<http://reddit.com/r/worldnews>

((lemma PRED >[accuse,arrest,clash,condemn,kill,slam,warn]/P.{so,x})
SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS])

Pattern 7.: Original conflict pattern (rewritten)

Wildcard Conflict Patterns

Replacing either the trigger sub-pattern, the predicate sub-pattern or both of them with a *semsim* function are the options for utilizing NESS-SHPM in a modified version of pattern 6 without modifying the general structure of the pattern. To evaluate which of these options are best suited to evaluate the retrieval performance of NESS-SHPM, three *wildcard conflict patterns* are constructed. In these patterns the predicate sub-pattern (pattern 8) or the trigger sub-pattern (pattern 9) are replaced by the wildcard operator.

(PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]) ^
(PRED/P */P)

Pattern 8.: Predicate wildcard pattern

(PRED/P.so,x SOURCE/C TARGET/C */T TOPIC/[RS]) ^
(lemma/J >PRED/P [accuse,arrest,clash,condemn,kill,slam,warn]/P)

Pattern 9.: Trigger wildcard pattern

Preliminary Evaluation The three wildcard conflict patterns are matched against the Worldnews-SH and the number of matches is recorded. Comparing the number of matches of these patterns shows which of the sub-patterns is most influential for the retrieval performance of pattern 6. Table 6.1 shows the results of these preliminary evaluations as well as the number of matches that result from matching pattern 6 against the Worldnes-SH. It can be seen that the choice of conflict verbs is much more influential on the number of matches than the choice of conflict prepositions when compared to the number of matches resulting from the original conflict pattern. While replacing the predicate sub-pattern with a wildcard operator yields an increase with a factor of 12,45, replacing the trigger sub-pattern with a wildcard operator only yields an increase with a factor of 1,07.

SemSim Conflict Patterns

Based on the result of the preliminary evaluation in section 6.1.3, the predicate sub-pattern of pattern 6 is replaced by different forms of *semsim* functional patterns to construct different *semsim conflict patterns*. These patterns are then used to evaluate the effects of utilizing NESS-SHPM. The trigger sub-pattern is not modified to better isolate these effects in comparison to purely symbolic SHPM.

Pattern 10 describes the general form of a *semsim* conflict pattern. The <SEMSIM-FUNCTION> placeholder is replaced with one of the three implemented *semsim* functions to construct the *semsim-fix conflict pattern* (pattern 11), *semsim-fix-lemma conflict pattern* (pattern 12) and the *semsim-ctx conflict pattern* (pattern 13). As <SEMSIM-ARGUMENT> the conflict verb list is used as similarity reference words in pattern 11 and pattern 12, which utilize FNESS. In the *semsim-ctx* conflict pattern, the wildcard operator is used as <SEMSIM-ARGUMENT>

Pattern name	Number of matches
Original conflict pattern	5766
Predicate wildcard pattern	71804
Trigger wildcard pattern	6154

Table 6.1.: Results of matching the wildcard patterns against the Worldnews-SH

since the necessary reference edges can only be provided via an external parameter and not inside the pattern.

$$(\text{PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]}) \wedge$$

$$(\text{<SEMSIM-FUNCTION>/J PRED/P <SEMSIM-ARGUMENT>/P})$$

Pattern 10.: General SemSim conflict pattern

$$(\text{PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]}) \wedge$$

$$(\text{semsim/J PRED/P [accuse,arrest,clash,condemn,kill,slam,warn]//P})$$

Pattern 11.: semsim-fix conflict pattern

$$(\text{PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]}) \wedge$$

$$(\text{semsim-fix-lemma/J PRED/P [accuse,arrest,clash,condemn,kill,slam,warn]//P})$$

Pattern 12.: semsim-fix-lemma conflict pattern

$$(\text{PRED/P.so,x SOURCE/C TARGET/C [against,for,of,over]/T TOPIC/[RS]}) \wedge$$

$$(\text{semsim-ctx/J PRED/P */P})$$

Pattern 13.: semsim-ctx conflict pattern

6.2. Conflict Dataset

To conduct an evaluation which assesses the retrieval performance of the NESS-SHPM system it is necessary to have a dataset of edges with labels that state whether an edge is an expression of conflict or not. Since such a dataset does not exist it needs to be constructed. In the following the construction process of this *conflict dataset* (CD), which is used for the evaluation in this case study, and the datasets characteristics are discussed.

6.2.1. Base Edge Set

The set of edges that can be retrieved by a conflict pattern, i.e. the original conflict pattern or a semsim conflict pattern is restricted the general form of these patterns. This entails that, given the same SH, every set of matching edges of a pattern of this form will be a subset of the matching edges of the predicate wildcard pattern (pattern 8). The set of edges resulting from matching this pattern against the Worldnews-SH are therefore used as the *base edge set* (BES) from which the conflict dataset is constructed, instead of the entirety of all the hypergraphs root edges.

Predicate Lemma Every edge in the BES has a predicate sub-edge that has an innermost atom, which is a verb that has a lemma. In the following this is called the *predicate lemma* of an edge. Each of the edges matching pattern 6 or a pattern in the form of of pattern 10 therefore corresponds to a predicate lemma.

6.2.2. Desired Characteristics

To effectively evaluate the effectiveness of the application of NESS by matching a pattern in the form of pattern 10, the dataset used for this should have the following characteristics:

- Contain the largest possible number of unique predicate lemmas
- Contain the largest possible number of edges per unique predicate lemma

On the one hand it is desired to have as many different unique predicate lemmas as possible in the dataset to be able to evaluate whether NESS can differentiate if a predicate lemma indicates an expression of conflict or not. On the other hand it is desired to have as many different edges per unique lemma as possible in the dataset to be able to evaluate whether CNESS is able to differentiate if edges represent an expression of conflict or not, given that they correspond to the same predicate lemma.

6.2.3. Construction Process

To create the labelled CD, the edges of the dataset need to be manually labelled by human annotators, which is labor-intensive. The BES contains $n_b = 71804$ edges. Due to the time constraints of this work and the limited availability of three annotators, the BES needs to be subsampled to create the CD.

Filtering

Since the desired characteristics described above relate the the distribution of predicate lemmas, it is relevant to verify that is possible to determine the predicate lemma for all edges in the edge set from which the CD is sampled. In some cases it is not possible to determine the predicate lemma of a given edge due to to implementation issues, which out of scope of this work. In these cases an edge is filtered from the BES, which results in the *filtered base edge set* (FBES). The FBES contains $n_f = 69380$ edges.

Sampling

The edges in the FBES correspond to $n_l = 2195$ unique predicate lemmas. Attaining to the desired dataset characteristics, the number of samples n_s in the subsampled dataset should ideally be a multiple $m_l \geq 2$ of n_l , so that $n_s = m_l \cdot n_l$. This would mean that every predicate lemma contained in the FBES is statistically represented multiple times in the subsampled dataset.

A dataset size of $n_s = 2000$ was chosen, wich means $m_l < 2$ and $n_s \ll n_f$. This entails that a trade-off between the desired dataset characteristics has to be made. To account for this, a sampling method is applied that offers more control over the distribution of

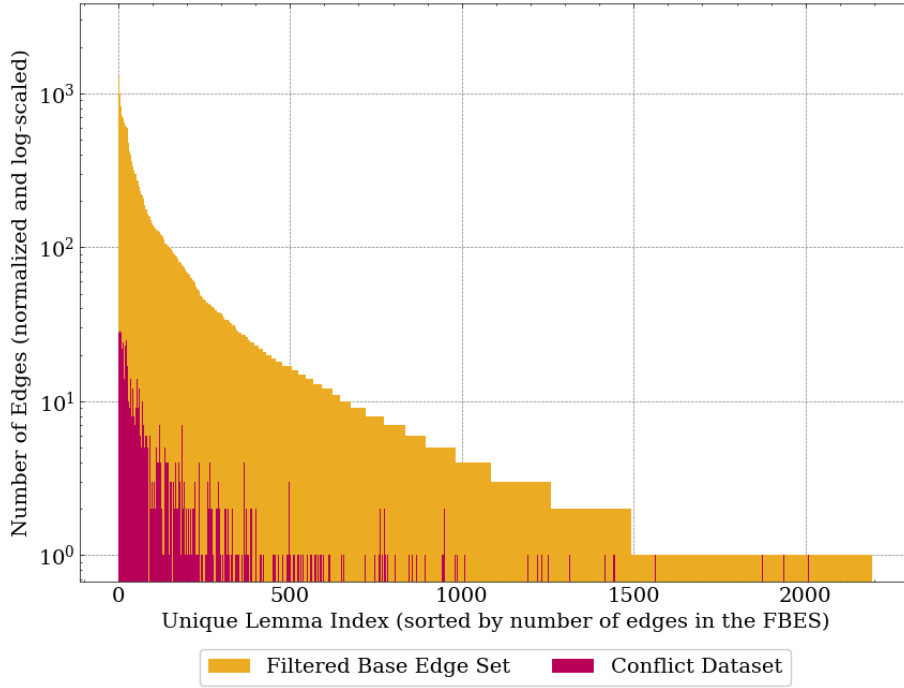


Figure 6.1.: Distribution of unique lemmas in the FBES and CD

predicate lemmas in the subsampled dataset than uniform random sampling does. This sampling method is based on the idea of *Stratified Sampling* (Parsons 2017) and is described in detail in algorithm 1.

is this correct?

The procedure splits the FBES into multiple bins after the edges are sorted by number of occurrence of their predicate lemma and then uniformly randomly samples from each bin. This method guarantees that predicate lemmas which correspond to a relatively small number of edges in the FBES will be represented in the subsampled dataset. The distribution of unique lemmas in the FBES and the CD is compared visually in fig. 6.1.

Algorithm 1 Dataset sampling algorithm

1. Create a list of tuples t of edges and their corresponding predicate lemma:
 $L = [(l_k, e_i), \dots]$ with $k \in \{0, \dots, m\}$ and $i \in \{0, \dots, n\}$
 2. Sort this list by the number of tuples containing a predicate lemma to create the list:
 $L_{sort} = [(l_0, e_0), \dots, (l_m, e_n)]$, so that:
 - n_k is the number of tuples containing a lemma l_k
 - t_j with $j > i$ is a tuple with sorted after tuple t_i
 - $n_o \geq n_p$ if $t_i = (l_o, e_i)$ and $t_j = (l_p, e_j)$
 3. Split the list L_{sort} into n_b bins.
 4. Uniformly sample n_{sb} tuples from each bin.
 5. Build a set of all edges e contained in the sampled tuples.
-

The subsampled dataset size resulting from this sampling method is $n_s = n_b * n_{sb}$. Given

Edge set name	Number of all edges	Number of un. lemmas	Number of conflict edges (% of all edges)	Number of no conflict edges (% of all edges)
Worldnews-SH	479384	-	-	-
Base Edge Set (BES)	71804	-	-	-
Filtered BES (FBES)	69380	2195	-	-
Conflict Dataset (CD)	2000	539	599 (29.95 %)	1401 (70.05 %)

Table 6.2.: Number of edges, number of unique lemmas and proportion of labels for the different edge sets

$n_s = 2000$, the values $n_b = 10$ and $n_{sb} = 200$ were chosen for sampling the CD.

Labelling

The labelling task is shared between the three annotators. A given edge will be either labeled as *conflict* or *no conflict* by an annotator following the definition given in section 6.1.1. Because of the aforementioned time constraints, every edge is only labeled by one annotator. To nonetheless ensure a consistent labelling among all annotators, a set of 50 edge is labelled by all three annotators. Every edge for which a disagreement in labelling occurs between at least two of the annotators, is inspected to reach an agreement on the label. Utilizing this process, the annotators understanding of what constitutes an expression of conflict is refined. Following this preliminary step, the n_s edges of the dataset are equally distributed among the three annotators and individually labelled by them.

6.2.4. Edge Set Comparison

The CD is the result of the filtering, sampling and labelling described above. The size of the Worldnews-SH, BES, FBES and CD are listed in table 6.2 for comparison. If applicable the the number of unique lemmas as well as the number and percentage of edges which are labelled as an expression of conflict and of those which are not are also noted.

add ex-
amples?
(in ap-
pendix?)

6.3. Evaluation Process

In this evaluation multiple *evaluation runs* are conducted. Each evaluation run correspond to a SHPM process in which a pattern is matched against the CD. In the case of patterns utilizing NESS this requires that additional parameters in form of an *NESS configuration* are given to the matching process. An evaluation run is described by an *evaluation (run) configuration*. For each evaluation run the *evaluation metrics* are computed.

6.3.1. Evaluation Run Configurations

An evaluation configuration consists of the following parameters:

- Conflict Pattern

add refer-
ences to
chapter 4

- NESS Configuration (in case NESS-SHMP):
 - NESS model
 - Similarity Threshold
 - Use all tokens (in the case of CNESS)
 - Reference Edge Set (in the case of CNESS)

Conflict Patterns The four conflict patterns used in this evaluation are described in detail in section 6.1.3. An overview of the properties of these patterns can be seen in table 6.3.

Similarity Thresholds In this evaluation the similarity threshold t_s is always selected from a range of thresholds $r_t = \{0, 0.01, \dots, 0.99, 1.00\}$, i.e. $t_s \in r_t$. This results in 101 different values of t_s .

Reference Edge Sets Multiple *reference edge sets* (RES) are randomly sampled from the set of edges in the CD, which are labelled as "conflict". These edges are then excluded from the dataset, to avoid introducing data from the test dataset to the system that is being evaluated. To compare the effect of different sample sizes, differently sized sets are drawn. To compare the effect of different samples, different samples are drawn. A RES with ID $N-X$ has $N \in \{1, 3, 10\}$ samples and is from sample draw $X \in \{1, 2, 3, 4, 5\}$. This results in 15 different RES in total. The specific sets that have been sampled can be seen in appendix A.1.

Evaluation Run Names

An evaluation run name has the form: **CP NM-AT r-N-X t-TS**

In a specific evaluation run name, the placeholders (capitalised letters) are replaced with actual values. Such a name always begins with the conflict pattern (CP) name in its shortened form: *original*, *semsim-fix*, *semsim-fix-lemma* or *semsim-ctx*. In case of a NESS utilizing conflict pattern, the NESS model (NM) name is added in its shortened form: word2vec as *w2v* and conceptnet-numberbatch as *cn*. If the NESS type is CNESS, the usage of the all-tokens (AT) option is indicated by adding *-at* to the model name. If the option is not used, it is not added. Also in case of CNESS, the reference edge set ID $N-X$ is indicated by appending *r-N-X*. For all NESS utilizing evaluation runs, the similarity threshold t_s is indicated by appending *t-TS*, where **TS** is the value of t_s .

Specification of Evaluation Run Configurations

The configurations for all evaluation runs that are conducted in this case study are specified in table 6.4. All possible parameter combinations of an evaluation configuration, i.e. the conflict pattern and the NESS parameters, are evaluated. The total number of conducted evaluation runs therefore amounts to 6465.² In table 6.4 the different values for the

²1 (original) + 2 (semsim-fix and semxim-fix-lemma) * 2 (FNESS models) * 101 (STs)
+ 1 (semsim-ctx) * 2 (CNESS models) * 2 (all tokens) * 15 (ref. edge sets) * 101 (STs)

Pattern name	Lemma based	NESS type	Includes ref. words	Requires ref. edges
Original conflict pattern (6)	Yes	-	-	-
sensim-fix conflict pattern (11)	No	Fixed	Yes	No
sensim-fix-lemma conflict (12)	Yes	Fixed	Yes	No
sensim-ctx conflict patter (13)	No	Contextual	No	Yes

Table 6.3.: Properties of the conflict patterns used in the evaluation

reference edge set ID and the ST are omitted. The *random* evaluation run configuration relates to a hypothetical evaluation run in which edges are uniformly randomly matched.

6.3.2. Evaluation Metrics

Using the information provided by the dataset labels it is determined whether a match is correct or not. If an edge matches in a given evaluation run and is labeled "conflict" in the dataset, it is considered a *true positive* (TP). If an edge matches but is labeled "no conflict", it is considered a *false positive* (FP). The *true negatives* (TN) and *false negatives* (FN) are determined analogously by examining the non-matching edges. Based on the TP, FP, TN and FN the metrics *precision*, *recall* and *F1-score* are computed.

Relationship of Similarity Threshold and Recall It can be generally stated that the recall (r) of NESS-SHMP in relation to the similarity threshold ($r(t_s)$) is strictly monotonically decreasing, since the set of points in embedding space that is inside of the similarity boundary consistently gets smaller with increasing threshold.

derive why these metrics were chosen:

accuracy is not interesting since the dataset is unbalanced, precision and recall both of interest, but are expected to be a trade-off (where recall should decline with rising ST). F1-score is an established metric that closely relates to precision and recall and represents this trade-off and therefore retrieval performance as a whole. MCC is arguably a better metric because it is symmetrical and incorporates true negatives. also the F1-score of the original pattern is worse than random, which indicates a metric mismatch. then again the close relation and equal value range of precision, recall and F1-score are a plus (for plotting especially).

show how this metrics are computed?

maybe add ref. to earlier section

6.4. Evaluation Results

In this section the results of the evaluation runs which are defined by the evaluation configurations in section 6.3.1 are examined. Different perspectives on the result data are constructed in the form of tables and plots to enable answering the research questions. The following subsections each represent one perspective and conclude with significant observations that can be made based on it.

Evaluation Run Name	Conflict Pattern	NESS Configuration	
		NESS Model	all tokens
original	original	-	-
semsim-fix w2v	semsim-fix	word2vec	-
semsim-fix cn	semsim-fix	conceptnet-numbatch	-
semsim-fix-lemma w2v	semsim-fix-lemma	word2vec	-
semsim-fix-lemma cn	semsim-fix-lemma	conceptnet-numbatch	-
semsim-ctx e5 r-N-X	semsim-ctx	e5	No
semsim-ctx gte r-N-X	semsim-ctx	gte	No
semsim-ctx e5-at r-N-X	semsim-ctx	e5	Yes
semsim-ctx gte-at r-N-X	semsim-ctx	gte	Yes

Table 6.4.: Evaluation Run Configurations

Result Data Description Concepts

To facilitate constructing insightful perspectives on the result data, some novel concepts for its description are introduced in the following.

Evaluation Run Sets Multiple evaluation runs can be grouped into an *evaluation run set* (ERS) according to their shared configuration parameter values. The naming convention for an ERS follows the evaluation run naming convention described in section 6.3.1. The parameters values that are not shared among the evaluation runs in the ERS are omitted from the name or replaced by the wildcard symbol *. The placeholders (capitalised letters) are used to refer to an ESR of a generic form with fixed parameter values without specifying these values. By surrounding a part of the ERS name with parentheses (*), it is indicated that this part is omitted if unsuitable.

Examples are given to illustrate this:

- An ERS of all evaluation runs utilizing NESS with $t_s = 0.5$ is named:
`semsim-* t-0.5`
- An ERS of all evaluation runs utilizing FNESS with an unspecified but fixed NESS model has the form:
`semsim-fix(-*) NM`
- An ERS of all evaluation runs utilizing NESS with all parameters (that are applicable) fixed but unspecified, except for the specific value $t_s = 0.5$, has the form:
`semsim-* NM(-AT) (r-N-X) t-0.5`

Best F1-Score Evaluation run The *best F1-Score evaluation run* refers to the evaluation run with the highest F1-Score in an ERS corresponding to a NESS utilizing evaluation configuration where every parameter except for t_s is fixed. Such an ERS is generally named `semsim-* NM(-AT) (r-N-X)`. The corresponding F1-score is also simply referred to as *best F1-score*.

Mean Reference Edge Set Evaluation Runs A *mean reference edge set evaluation run* is constructed from the mean value of all evaluation scores for the evaluation runs in an ERS of the form `semsim-ctx NM-AT r-N-* t-TS`. This means for every t_s the mean of the corresponding evaluation scores of all reference edge sets of the same size is computed. In the following these synthetical evaluation runs are referred to in this form: `semsim-ctx NM-AT r-N-mean`

Mean Reference Edge Set Best F1-Score Evaluation Metric Scores The *mean reference edge set (RES) best F1-Score evaluation metric scores* are the mean values of all evaluation scores corresponding to the best F1 score for all evaluation runs in an ERS of the form `semsim-ctx NM-AT r-N-*`. In the following these evaluation metric scores will be referred to in this form: `semsim-ctx NM-AT r-N-mean-best`

6.4.1. Best F1-Score based Evaluation Run Comparison

Table 6.5 shows the evaluation scores for all evaluation metrics of the best F1-score evaluation runs for the original conflict pattern evaluation run and all evaluation runs utilizing FNESS. For the evaluation runs utilizing CNESS only the mean RES best F1-score evaluation metric scores and the standard deviation of the best F1-scores for the corresponding ERSs are shown. The t_s value listed for the mean RES best F1 score evaluation metrics is the mean of all t_s values for the best F1-scores in the corresponding ERSs.

In table A.5 and table A.6 of appendix A.2 the best F1-score evaluation run results can be seen for evaluation runs utilizing CNESS with all tokens disabled and enabled respectively. These tables also list the hypothetical best F1-score evaluation for run the mean reference edge set evaluation runs. Additionally the mean standard deviation for these ERSs is shown, i.e. the mean of the standard deviations of the F1-score for every ERS of the form `semsim-ctx NM-AT r-N-* t-TS`.

Significant Observations

- 1.1 All evaluation runs utilizing NESS achieve a best F1-score that is higher than the F1-score of the random evaluation run and the original evaluation run
- 1.2 CNESS achieves a higher F1-score than FNESS by 4.0%, when comparing the highest F1-scores achieved among all FNESS utilizing evaluation runs and the highest mean RES best F1 score achieved among all CNESS utilizing evaluation runs (`semsim-fix-lemma cn t-0.30` and `semsim-ctx e5 r-10-mean-best`)
- 1.3 Lemma based FNESS achieves a higher F1-score than non-lemma FNESS by 3.2%, when comparing the highest F1-scores achieved by evaluation runs utilizing one of the two variants (`semsim-fix w2v t-0.27` and `semsim-fix-lemma cn t-0.30`)
- 1.4 For lemma based FNESS, the conceptnet-numberbatch model achieves a higher best F1-score than the word2vec model by 4.2% (`semsim-fix-lemma cn t-0.30` vs `semsim-fix-lemma w2v t-0.33`)
- 1.5 For not lemma based FNESS, the word2vec model achieves a higher best F1-score than the conceptnet-numberbatch model by 1.4% (`semsim-fix w2v t-0.27` vs `semsim-fix cn t-0.25`)

Evaluation Run Name				Prec.	Rec.	(Best) F1-Score	
CP	NM	RES	t_s				Std. Dev.
random			-	0.300	0.500	0.375	-
original			-	0.706	0.209	0.322	-
semsim-fix	cn		0.25	0.479	0.524	0.500	-
semsim-fix	w2v		0.27	0.483	0.533	0.507	-
semsim-fix-l.	cn		0.30	0.492	0.558	0.523	-
semsim-fix-l.	w2v		0.33	0.460	0.553	0.502	-
semsim-ctx	e5	r-1-mean-best	0.65	0.392	0.772	0.518	+/- 0.025
semsim-ctx	gte	r-1-mean-best	0.59	0.336	0.879	0.483	+/- 0.025
semsim-ctx	e5	r-3-mean-best	0.68	0.399	0.818	0.536	+/- 0.021
semsim-ctx	gte	r-3-mean-best	0.65	0.365	0.799	0.499	+/- 0.016
semsim-ctx	e5	r-10-mean-best	0.72	0.416	0.790	0.544	+/- 0.020
semsim-ctx	gte	r-10-mean-best	0.68	0.382	0.812	0.517	+/- 0.010
semsim-ctx	e5-at	r-1-mean-best	0.69	0.369	0.841	0.509	+/- 0.016
semsim-ctx	gte-at	r-1-mean-best	0.66	0.335	0.882	0.483	+/- 0.021
semsim-ctx	e5-at	r-3-mean-best	0.72	0.378	0.821	0.516	+/- 0.011
semsim-ctx	gte-at	r-3-mean-best	0.70	0.336	0.876	0.485	+/- 0.017
semsim-ctx	e5-at	r-10-mean-best	0.74	0.382	0.843	0.525	+/- 0.012
semsim-ctx	gte-at	r-10-mean-best	0.72	0.338	0.900	0.491	+/- 0.008

Table 6.5.: Evaluation scores corresponding to best F1-scores for all evaluation runs

- 1.6 CNESS with the AT option disabled achieves a higher or equal mean RES best F1-score than CNESS with AT option enabled in 6/6 (100%) direct comparisons
(semsim-ctx NM r-N-mean-best vs semsim-ctx NM-at r-N-mean-best)
- 1.7 CNESS with the e5 model achieves a higher or equal mean RES best F1-score than CNESS with the gte model in 6/6 (100%) direct comparisons
(semsim-ctx e5-* r-N-mean-best vs semsim-ctx gte-* r-N-mean-best)
- 1.8 CNESS with the AT option enabled has a lower standard deviation of best F1-score than CNESS with the AT option disabled in 5/6 (83%) direct comparisons
(semsim-ctx NM-at r-N-mean-best vs semsim-ctx NM r-N-mean-best)

6.4.2. Evaluation Metric vs. Similarity Threshold

These plots visualise the resulting evaluation scores for the different evaluation metrics in relation to different values for the similarity threshold.

Figure 6.2a shows the F1-score vs. the ST for the best performing evaluation runs for each conflict pattern. That means for every conflict pattern, this shows the evaluation run(s) with the configuration that resulted in the highest best F1-score. For the **random** and **original** evaluation runs, there is obviously no configuration to choose from.

For the FNESS utilizing evaluation runs, the evaluation run with the highest F1-score in the ERSs of the form `semsim-fix(-lemma)` NM is selected. This means for the `semsim-fix` and `semxim-fix-lemma` conflict patterns the corresponding best F1-score evaluation runs of the best performing model are selected..

For the CNESS utilizing evaluation runs, the evaluation runs which correspond to highest mean RES best F1-score are selected. This means the ERS of the form **semsim-ctx NM-AT r-N-*** which resulted in the highest mean value of best F1-scores. This ERS consists of five evaluation runs, which each have the form **semsim-ctx NM-AT r-N-X**. The F1-scores for these runs are plotted with a lighter curve. The additional synthetic **semsim-ctx NM-AT r-N-mean** score is plotted with a normally bold curve.

Figure 6.2b follows the same concept as fig. 6.2a, but instead of the F1-score this plot shows the scores of precision and recall vs. the ST. Also in the selection of evaluation runs, the **semsim-fix** (not lemma) conflict pattern based evaluation runs are excluded.

The plots in this section are selected because they are deemed to be most relevant for the following. A more comprehensive comparison of the different evaluation runs from this perspective can be found in appendix A.3.

Active Similarity Threshold Range To facilitate the description of observation for this perspective on the result data, the concept of the *active similarity threshold range* (ASTR) is introduced. For a given ESR of the form **semsim-* NM(-AT) (r-N-X)**, the ASTR describes the range of t_s for which the recall (r) that is achieved by these evaluation runs is $r \neq r_{max} = 1$ and $r \neq r_{min}$. Here r_{min} and r_{max} are the lowest and highest recall values, which correspond to $t_{s1} \leq t_{s2}$, since the function $r(t_s)$ is monotonically decreasing.

Significant Observations

- 2.1 The ASTR of **semsim-fix-lemma cn** is larger ($0.0 < t_s < 1.0$) than the ASTR of **semsim-ctx e5** (ca. $0.625 < t_s < 0.875$)
- 2.2 Generally the ASTR of ERSs of the form **semsim-fix-* NM** is larger than the the ASTR of ERSs of the form **semsim-ctx** (confer also fig. A.1, fig. A.2 fig. A.5 and fig. A.6)
- 2.3 The ASTRs are nearly equal for ERSs of the form **semsim-fix-* NM** (confer fig. A.1 and fig. A.2)
- 2.4 The ASTR of ERSs of the form **semsim-ctx gte-AT** begin at a lower value than for ERSs of the form **semsim-ctx e5-AT** (confer fig. A.5 and fig. A.4)
- 2.5 The ASTR of ERSs of the form **semsim-ctx NM** begin at a lower value than for ERSs of the form **semsim-ctx NM-at** (confer fig. A.6 and fig. A.4)
- 2.6 The ASTRs end at nearly the same value for ERSs of the form **semsim-ctx NM-AT** (confer fig. A.5, fig. A.6 and fig. A.4)
- 2.7 The evaluation runs in the ERS **semsim-fix-lemma cn** achieve a higher F1 value than the evaluation run in the ERS **semsim-fix w2v** for nearly all values of t_s
- 2.8 The precision of the evaluation run which achieves the highest precision among those in the synthetic ESR **semsim-ctx e5 r-10-mean** (p_{max}) and the precision of the original evaluation run (p_{og}) are nearly equal ($p_{max} \approx p_{og}$)
- 2.9 The precision of evaluation run **semsim-fix-lemma cn** correlates with the ST until it reaches the value achieved by the **original** evaluation run, where it plateaus

- 2.10 The precision of the evaluation runs in ERS `semsim-ctx e5` correlate with the ST until precision (p) and recall (r) reach approximately the same value ($p \approx r \approx 0.5$), after which it fluctuates (the specific fluctuation varies with the specific RES)
- 2.11 The precision achieved by evaluation runs in ERSs of the form `semsim-ctx e5 r-10-*` t-TS has a higher variation for $t_s \geq 0.75$ than for $t_s < 0.75$
- 2.12 The precision of the evaluation run which achieves the highest precision among those in the synthetic ESR `semsim-ctx e5 r-10-mean` (p_{\max}) and the precision of the original evaluation run (p_{og}) are nearly equal ($p_{\max} \approx p_{\text{og}}$)
- 2.13 The evaluation metric scores of `semsim-fix-lemma cn t-1.00` are lower than those of the `original` evaluation run

should i quantify all these observations?

would a more detailed analysis of precision and recall make sense? maybe a precision-recall curve or an roc? maybe recall at best precision and precision at random recall?

6.4.3. Best F1-Score vs. Number of Reference Edges

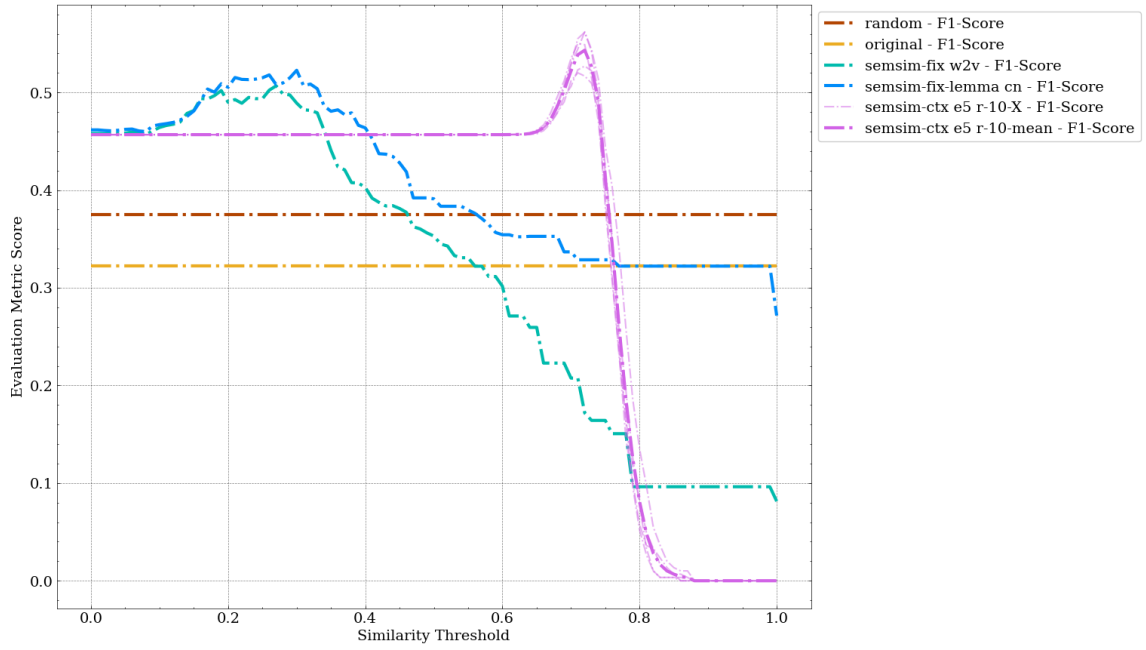
Figure 6.3 visualises the relation of the number of reference edges and the best-F1 score. For this purpose the number of reference edges N is plotted versus the mean RES best F1-score for all ERSs of the form `semsim-ctx NM-AT r-N-*`. The standard deviation of the best F1 scores for these ERSs is visualised by the shaded areas around the curves of the mean RES best F1-scores.

Significant Observations

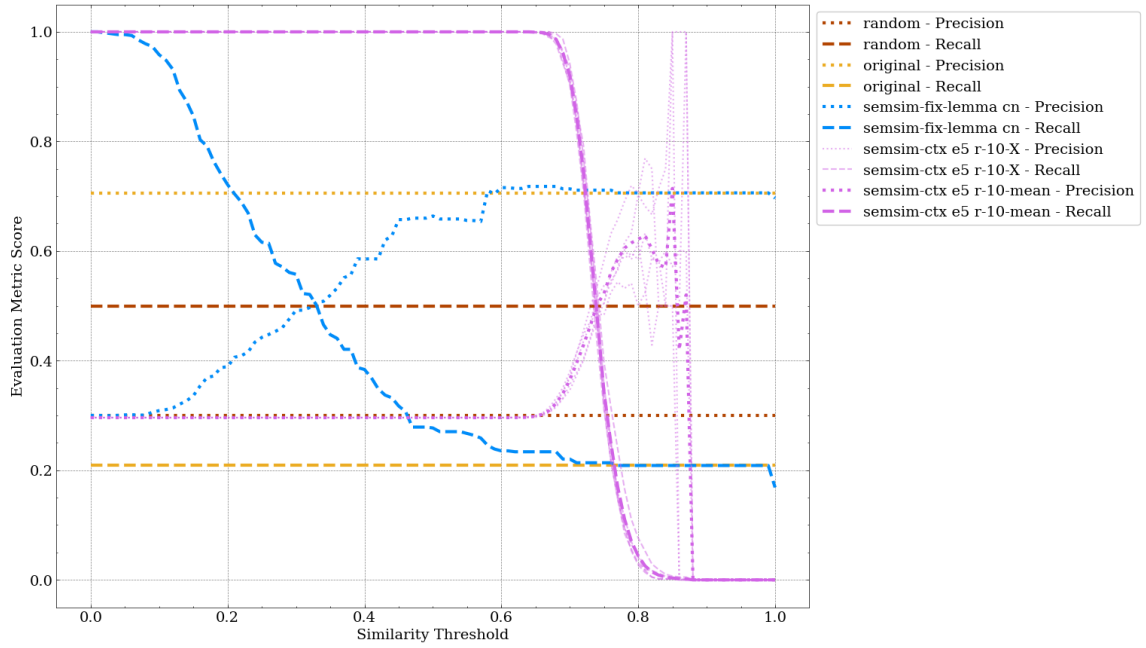
- 3.1 The mean RES best F1-score of the evaluation runs in an ERS of the form `semsim-ctx NM(-AT) r- N_1 -*` is higher than the mean RES best F1-score of the evaluation runs in an ERS of the form `semsim-ctx NM(-AT) r- N_2 -*`, if $N_2 > N_1$ for $N_1, N_2 \in \{1, 3, 10\}$
- 3.2 The standard deviation of the mean RES best F1-score of the evaluation runs in an ERS of the form `semsim-ctx NM(-AT) r- N_1 -*` is lower than the standard deviation of the mean RES best F1-score of the evaluation runs in an ERS of the form `semsim-ctx NM(-AT) r- N_2 -*`, if $N_2 > N_1$ for $N_1, N_2 \in \{1, 3, 10\}$, except for `semsim-ctx et-at r-3-*` ($sd_1 = 0.11$) and `semsim-ctx et-at r-10-*` ($sd_2 = 0.12$)

6.4.4. Predicate Lemma based Evaluation Run Comparison

In this section it is explored how the different NESS systems differ in which edges they match. This is done by following up on the concept of the predicate lemma introduced in section 6.2.1. In section 6.2.2 one of the two desired characteristics of the dataset states that it should contain the largest possible number of edges per unique predicate lemma. Specifically it is of interest, how the CNESS system performs in comparison to the FNESS



(a) F1-score vs. ST for the evaluation runs `random`, `original`, `semsim-fix w2v`, `semsim-fix-lemma cn` and `semsim-ctx e5 r-10-X`



(b) Precision and recall vs. ST for the evaluation runs `random`, `original`, `semsim-fix-lemma cn` and `semsim-ctx e5 r-10-X`

Figure 6.2.: Evaluation metric scores vs. similarity threshold values

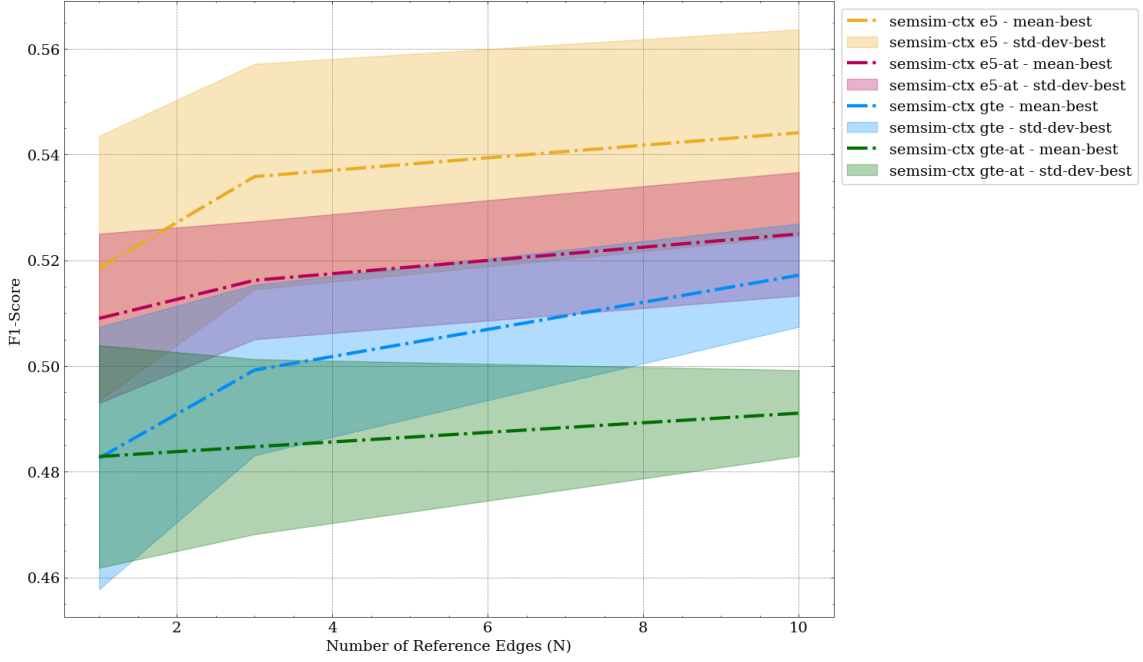


Figure 6.3.: Mean reference edge set best F1-score for the ERSs `semsim-ctx e5 r-N-*`, `semsim-ctx e5-at r-N-*`, `semsim-ctx gte-at r-N-*` and `semsim-ctx gte-at r-N-*`

system for subsets of edges which share the same predicate lemma. Such a subset of edges of the conflict dataset is in the following referred to as *predicate lemma edge set* (PLES).

NESS Type Representatives Two evaluation runs are selected to represent the two versions of the NESS systems for the comparison. The lemma-based version of FNESS is chosen here, because of its superior performance regarding best F1-score that is observed in section 6.4.1 and section 6.4.2. Specifically, the evaluation runs `semsim-fix-lemma cn t-0.30` (evaluation run *A*) and `semsim-ctx e5 r-10-2 t-0.72` (evaluation run *B*) are selected as representatives of the FNESS and CNESS system respectively. These are the best performing evaluation runs regarding F1-score for the respective semsim conflict patterns (i.e. NESS system), which can be seen in table 6.5 and table A.5.³

PLES Evaluation Score based Evaluation Run Comparison

Label Balance Ratio The *label balance ratio* (LBR) measures how balanced the labels in a given set of labelled edges are. It is calculated by eq. (6.1). Here n_{pos} and n_{neg} are the number of positively ("conflict") and negatively ("no conflict") labeled edges in the edge set. An edge set with fully balanced labels has $LBR = 1$ and completely unbalanced labeled edge set has $LBR = 0$.

³The latter table specifically shows that the evaluation runs `semsim-ctx e5 r-1-2 t-0.67`, `semsim-ctx e5 r-10-2 t-0.72` and `semsim-ctx e5 r-10-4 t-0.72` all correspond to an F1 score $s_{F1} = 0.56$. The evaluation runs utilizing the RESs of size $N = 10$ are chosen over the one utilizing an RES of size $N = 1$, because of their generally superior performance regarding F1 score, which is observed in section 6.4.3. Among the two remaining evaluation runs, `semsim-ctx e5 r-10-2 t-0.72` is selected randomly.

$$LBR = 1 - \left(\frac{|n_{\text{pos}} - n_{\text{neg}}|}{n_{\text{pos}} + n_{\text{neg}}} \right) \quad (6.1)$$

The evaluation metrics are computed for evaluation run A and B for each PLES, along with metrics that measure distributional properties of a PLES. Namely the number of edges n_e , the number of positively labeled and negatively labeled edges (n_{pos} and n_{neg}), the LBR and the entropy of a PLES.

Table 6.6 lists the ten predicate lemmas for whose PLES the absolute difference in F1-score, which is achieved in the two evaluation runs, is the highest. Conversely table 6.8 lists the ten predicate lemma for whose PLES the difference in F1-score which is achieved in the tow evaluation runs is the lowest. In both tables the predicate lemmas have been filtered beforehand, so that only PLES with $n_s \geq 5$ samples are considered. Additionally the recalls (r_A, r_B) achieved by both evaluation runs regarding a PLES must fulfil the condition that $r_A + r_B > 0$, i.e. at least the recall achieved by one of the evaluation runs must be non-zero. Table 6.7 follows the same concept as table 6.6, except for the condition regarding the recalls being $r_A * r_B > 0$, i.e. both recalls achieved by the tow evaluation runs must be non-zero. This second variant of the table is not shown for table 6.8, because it only lists lemmas for whose PLES the F1-score achieved by both evaluation runs is zero.

should I add a table with the actual labels produced by the evaluation runs?

Significant Observations

- 4.1 The CNESS utilizing evaluation run achieves a higher F1-score than the FNESS utilizing evaluation run for every PLES of the the top ten PLES regarding highest difference in F1-score between the two evaluation runs (independently of the recall condition)
- 4.2 The mean LBR and mean entropy of the top ten PLESs regarding the highest difference in F1-score between the two evaluation runs are higher than the mean LBR and mean entropy of the top ten PLESs regarding the lowest difference in F1-score between the two evaluation runs (independently of the recall condition)
- 4.3 The differences in F1-score achieved by the two evaluation runs is higher for the recall condition $r_A + r_B > 0$ than for $r_A \cdot r_B > 0$, because for the first condition the F1-score of the FNESS evaluation run is zero for all PLESs
- 4.4 The lemmas corresponding to the the top ten PLES regarding highest difference in F1-score between the two evaluation runs are all not included in the conflict verbs: "accuse", "arrest", "clash", "condemn", "kill", "slam"
- 4.5 Of the lemmas corresponding to the top ten PLES regarding lowest difference in F1-score between the two evaluation runs, five of six are included in the conflict verbs: "arrest", "condemn", "kill", "slam" ("clash" is not included)

Lemma	F1 Diff.	F1 A	F1 B	n_e	$n_{\text{pos}}/n_{\text{neg}}$	LBR	Entropy
file	1.00	0.00	1.00	5	5/0	0.00	0.00
order	0.88	0.00	0.88	14	7/7	1.00	1.00
launch	0.86	0.00	0.86	14	8/6	0.86	0.99
step	0.80	0.00	0.80	5	2/3	0.80	0.97
target	0.77	0.00	0.77	8	6/2	0.50	0.81
use	0.75	0.00	0.75	14	5/9	0.71	0.94
block	0.73	0.00	0.73	8	4/4	1.00	1.00
take	0.71	0.00	0.71	19	6/13	0.63	0.90
open	0.67	0.00	0.67	8	1/7	0.25	0.54
suspend	0.67	0.00	0.67	6	2/4	0.67	0.92
build	0.67	0.00	0.67	6	1/5	0.33	0.65
						mean	
						0.61	0.79

Table 6.6.: Top ten lemmas regarding the highest difference in F1-score between the evaluation runs with recalls $r_A + r_B > 0$ and number of samples per lemma $n_s \geq 5$ for the evaluation runs `semsim-fix-lemma cn t-0.30` (A) and `semsim-ctx e5 r-10-2 t-0.72` (B)

Lemma	F1 Diff.	F1 A	F1 B	n_e	$n_{\text{pos}}/n_{\text{neg}}$	LBR	Entropy
accept	0.42	0.25	0.67	7	1/6	0.29	0.59
strike	0.30	0.80	0.50	9	6/3	0.67	0.92
capture	0.22	0.44	0.67	7	2/5	0.57	0.86
seize	0.17	0.67	0.50	12	6/6	1.00	1.00
deny	0.17	0.50	0.67	6	2/4	0.67	0.92
suggest	0.17	0.33	0.50	5	1/4	0.40	0.72
warn	0.12	0.93	0.81	24	21/3	0.25	0.54
claim	0.12	0.43	0.55	22	6/16	0.55	0.85
attack	0.09	0.91	1.00	12	10/2	0.33	0.65
threaten	0.09	0.52	0.61	20	7/13	0.70	0.93
approve	0.08	0.12	0.20	16	1/15	0.12	0.34
						mean	
						0.50	0.76

Table 6.7.: Top ten lemmas regarding the highest difference in F1-score between the evaluation runs with recalls $r_A \cdot r_B > 0$ and number of samples per lemma $n_s \geq 5$ for the evaluation runs `semsim-fix-lemma cn t-0.30` (A) and `semsim-ctx e5 r-10-2 t-0.72` (B)

Lemma	F1 Diff.	F1 A	F1 B	n_e	$n_{\text{pos}}/n_{\text{neg}}$	LBR	Entropy
arrest	0.00	1.00	1.00	11	11/0	0.00	0.00
slam	0.00	0.92	0.92	7	6/1	0.29	0.59
criticize	0.00	0.91	0.91	6	5/1	0.33	0.65
shoot	0.00	0.89	0.89	5	4/1	0.40	0.72
condemn	0.00	0.84	0.84	25	18/7	0.56	0.86
dismiss	0.00	0.80	0.80	6	4/2	0.67	0.92
tell	0.01	0.49	0.50	28	9/19	0.64	0.91
accuse	0.02	0.97	0.95	33	31/2	0.12	0.33
kill	0.02	0.66	0.64	77	38/39	0.99	1.00
say	0.03	0.45	0.48	31	9/22	0.58	0.87
						mean	
						0.46	0.68

Table 6.8.: Top ten lemmas regarding the lowest absolute difference in F1-score between the evaluation runs with recalls $r_A + r_B > 0$ and number of samples per lemma $n_s \geq 5$ for the evaluation runs `semsim-fix-lemma cn t-0.30` (A) and `semsim-ctx e5 r-10-2 t-0.72` (B)

6.5. Result Discussion

In this section the previously presented evaluation results are discussed. It synthesises the major insights derived from the observations of the different result data perspectives. The discussion is organised into categories and sub-categories, which relate to the research questions outlined in section 3.1. Here retrieval performance generally refers to joined measure of precision and recall and therefore means F1-score, as stated above in section 6.3.2.

Add statement about limitations to specific dataset and unknown generalisability

Missing observations and findings regarding "no breakpoints" (in recall or general?)

6.5.1. Retrieval Performance Improvement

- NESS-SHMP can achieve a better retrieval performance than the original conflict pattern, independent of NESS type and configuration (depending on the ST)
Supporting Observations: Item 1.1
- CNESS-SHPM using the sub tokens embedding can achieve the overall best retrieval performance (in comparison to FNESS-SHMP and the original conflict pattern)
Supporting Observations: Item 1.1, Item 1.2
- Using lemma-based FNESS-SHMP instead of word-based FNESS-SHMP can achieve a better retrieval performance
Supporting Observations: Item 1.3, Item 2.7

Similarity Threshold Impact

- The relation of ST and NESS-SHMP retrieval performance and therefore the relevant ST range depends primarily on the NESS type
Supporting Observations: Item 2.1 Item 2.2 Item 2.3
- The relation of ST and CNESS-SHMP retrieval performance depends secondarily on the NESS model and the usage of the all tokens option
Supporting Observations: Item 2.4 Item 2.5 Item 2.6

NESS Configuration Impact

- Using a generally better performing NESS model (regarding established benchmarks) does not generally improve the NESS-SHMP retrieval performance
Supporting Observations: Item 1.4, Item 1.5, Item 1.7
- Using the sub tokens embedding instead of the all tokens embedding improves retrieval performance, but using the all tokens embedding makes it less sensible to the selection of the reference edges
Supporting Observations: Item 1.6, Item 1.8
- CNESS-SHMP retrieval performance improves with a higher number of reference edges and is less sensible to the specific selection of reference edges
Supporting Observations: Item 3.1, Item 3.2

6.5.2. Retrieval Precision Behaviour

- The precision of NESS-SHMPM correlates with the ST until a specific value of the ST, which itself is specific to the NESS type, NESS model (and other CNESS parameters, especially the selection of reference edges)
Supporting Observations: Item 2.9, Item 2.10, Item 2.13
- CNESS-SHMP achieves on average the same precision as the original conflict pattern and lemma-based FNESS, although CNESS-SHMP can achieve a higher precision, it depends on the selection of the reference edges
Supporting Observations: Item 2.11, Item 2.12

6.5.3. Contextual Differentiation Ability

- CNESS-SHMP is able differentiate when matching a set of edges where purely symbolic SHPM and FNESS-SHMPM cannot, i.e. cases where context is needed to determine the correct semantics of word
Supporting Observations: Item 4.1, Item 4.2, Item 4.4, Item 4.5
- While CNESS-SHMP achieves a highest difference in retrieval performance for sets of edges, where FNESS-SHMP does not match, it also achieves a better retrieval performance in cases where FNESS-SHMP does match
Supporting Observations: Item 4.1 Item 4.3

Add or integrate more direct answer to the research questions

7. Related Work

8. Future Work

8.1. Conceptual Improvements

Somehow extend the token span used for CNESS beyond the word tokens but not to all tokens. Use the tokens of the next best sub-edge e.g. although in the case of a predicate this is probably the entire sentence most of the time.

different direction: employ knowledge based semantic similarity to maintain highest level of openness

8.2. Implementation Improvements

implemnt multiprocessing, i.e. server process for both hypergraph and semsim matchers.

other option would be to leverage python shared memory capabilities but is likely to be less stable and has less scaling potential

for CNESS, pre-compute embeddings for all possible edges (is this feasible?)

8.3. Further Evaluations

9. Conclusion

Bibliography

- Aizawa, Akiko (Jan. 1, 2003). “An Information-Theoretic Perspective of Tf-Idf Measures”. In: *Information Processing & Management* 39.1, pp. 45–65. ISSN: 0306-4573. DOI: 10.1016/S0306-4573(02)00021-3. URL: <https://www.sciencedirect.com/science/article/pii/S0306457302000213> (visited on 03/27/2024).
- Almeida, Felipe and Geraldo Xexéo (May 1, 2023). *Word Embeddings: A Survey*. DOI: 10.48550/arXiv.1901.09069. arXiv: 1901.09069 [cs, stat]. URL: <http://arxiv.org/abs/1901.09069> (visited on 07/26/2023). preprint.
- Auer, Sören et al. (2007). “DBpedia: A Nucleus for a Web of Open Data”. In: *The Semantic Web*. Ed. by Karl Aberer et al. Berlin, Heidelberg: Springer, pp. 722–735. ISBN: 978-3-540-76298-0. DOI: 10.1007/978-3-540-76298-0_52.
- Bengio, Yoshua, Réjean Ducharme, and Pascal Vincent (2000). “A Neural Probabilistic Language Model”. In: *Advances in Neural Information Processing Systems*. Vol. 13. MIT Press. URL: https://proceedings.neurips.cc/paper_files/paper/2000/hash/728f206c2a01bf572b5940d7d9a8fa4c-Abstract.html (visited on 04/04/2024).
- Bojanowski, Piotr et al. (2017). “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tac1_a_00051. URL: <https://aclanthology.org/Q17-1010> (visited on 07/31/2023).
- Bollegala, Danushka and James O’Neill (Apr. 1, 2022). *A Survey on Word Meta-Embedding Learning*. arXiv e-prints. DOI: 10.48550/arXiv.2204.11660. URL: <https://ui.adsabs.harvard.edu/abs/2022arXiv220411660B> (visited on 04/04/2024). preprint.
- Chandrasekaran, Dhivya and Vijay Mago (Feb. 18, 2021). “Evolution of Semantic Similarity—A Survey”. In: *ACM Computing Surveys* 54.2, 41:1–41:37. ISSN: 0360-0300. DOI: 10.1145/3440755. URL: <https://dl.acm.org/doi/10.1145/3440755> (visited on 06/17/2023).
- Chen, Stanley F. and Joshua Goodman (Oct. 1, 1999). “An Empirical Study of Smoothing Techniques for Language Modeling”. In: *Computer Speech & Language* 13.4, pp. 359–394. ISSN: 0885-2308. DOI: 10.1006/csla.1999.0128. URL: <https://www.sciencedirect.com/science/article/pii/S088523089901286> (visited on 04/04/2024).
- Chowdhary, K. R. (2020). “Natural Language Processing”. In: *Fundamentals of Artificial Intelligence*. Ed. by K.R. Chowdhary. New Delhi: Springer India, pp. 603–649. ISBN: 978-81-322-3972-7. DOI: 10.1007/978-81-322-3972-7_19. URL: https://doi.org/10.1007/978-81-322-3972-7_19 (visited on 03/05/2024).
- Church, Kenneth Ward and Patrick Hanks (June 1989). “Word Association Norms, Mutual Information, and Lexicography”. In: *27th Annual Meeting of the Association for Computational Linguistics*. ACL 1989. Vancouver, British Columbia, Canada: Association for Computational Linguistics, pp. 76–83. DOI: 10.3115/981623.981633. URL: <https://aclanthology.org/P89-1010> (visited on 03/29/2024).
- Collobert, Ronan and Jason Weston (July 5, 2008). “A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning”. In: *Proceedings of the 25th International Conference on Machine Learning*. ICML ’08. New York, NY, USA: Association for Computing Machinery, pp. 160–167. ISBN: 978-1-60558-205-4. DOI:

- 10.1145/1390156.1390177. URL: <https://dl.acm.org/doi/10.1145/1390156.1390177> (visited on 04/03/2024).
- Deerwester, Scott et al. (1990). "Indexing by Latent Semantic Analysis". In: *Journal of the American Society for Information Science* 41.6, pp. 391–407. ISSN: 1097-4571. DOI: 10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9> (visited on 03/29/2024).
- Evans, James A. and Pedro Aceves (July 1, 2016). *Machine Translation: Mining Text for Social Theory*. DOI: 10.1146/annurev-soc-081715-074206. URL: <https://papers.ssrn.com/abstract=2822747> (visited on 06/15/2023). preprint.
- Gabrilovich, Evgeniy, Shaul Markovitch, et al. (2007). "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis." In: *IJCAI*. Vol. 7, pp. 1606–1611.
- Ganitkevitch, Juri, Benjamin Van Durme, and Chris Callison-Burch (June 2013). "PPDB: The Paraphrase Database". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL-HLT 2013. Ed. by Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff. Atlanta, Georgia: Association for Computational Linguistics, pp. 758–764. URL: <https://aclanthology.org/N13-1092> (visited on 03/27/2024).
- García-Ferrero, Iker, Rodrigo Agerri, and German Rigau (Nov. 2021). "Benchmarking Meta-embeddings: What Works and What Does Not". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Findings 2021. Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 3957–3972. DOI: 10.18653/v1/2021.findings-emnlp.333. URL: <https://aclanthology.org/2021.findings-emnlp.333> (visited on 08/01/2023).
- Goikoetxea, Josu, Aitor Soroa, and Eneko Agirre (June 15, 2018). "Bilingual Embeddings with Random Walks over Multilingual Wordnets". In: *Knowledge-Based Systems* 150, pp. 218–230. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2018.03.017. URL: <https://www.sciencedirect.com/science/article/pii/S0950705118301412> (visited on 04/04/2024).
- Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (Nov. 10, 2016). *Deep Learning*. MIT Press. 801 pp. ISBN: 978-0-262-33737-3. Google Books: omivDQAAQBAJ.
- Han, Mengting et al. (2021). "A Survey on the Techniques, Applications, and Performance of Short Text Semantic Similarity". In: *Concurrency and Computation: Practice and Experience* 33.5, e5971. ISSN: 1532-0634. DOI: 10.1002/cpe.5971. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5971> (visited on 07/13/2023).
- Harispe, Sébastien et al. (2015). *Semantic Similarity from Natural Language and Ontology Analysis*. DOI: 10.2200/S00639ED1V01Y201504HLT027. arXiv: 1704.05295 [cs]. URL: <http://arxiv.org/abs/1704.05295> (visited on 06/19/2023).
- Harris, Zellig S. (Aug. 1, 1954). "Distributional Structure". In: *WORD* 10.2-3, pp. 146–162. ISSN: 0043-7956. DOI: 10.1080/00437956.1954.11659520. URL: <https://doi.org/10.1080/00437956.1954.11659520> (visited on 03/27/2024).
- He, Hua and Jimmy Lin (June 2016). "Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL-HLT 2016. Ed. by Kevin Knight, Ani Nenkova, and Owen Rambow. San Diego, California: Association for Computational Linguistics, pp. 937–948. DOI: 10.18653/v1/N16-1108. URL: <https://aclanthology.org/N16-1108> (visited on 04/01/2024).

- Hirschberg, Julia and Christopher D. Manning (July 17, 2015). “Advances in Natural Language Processing”. In: *Science* 349.6245, pp. 261–266. DOI: 10.1126/science.aaa8685. URL: <https://www.science.org/doi/abs/10.1126/science.aaa8685> (visited on 06/15/2023).
- Kowsari, Kamran et al. (Apr. 2019). “Text Classification Algorithms: A Survey”. In: *Information* 10.4 (4), p. 150. ISSN: 2078-2489. DOI: 10.3390/info10040150. URL: <https://www.mdpi.com/2078-2489/10/4/150> (visited on 03/06/2024).
- Landauer, Thomas K. and Susan T. Dumais (1997). “A Solution to Plato’s Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge”. In: *Psychological Review* 104.2, pp. 211–240. ISSN: 1939-1471. DOI: 10.1037/0033-295X.104.2.211.
- Landauer, Thomas K, Peter W. Foltz, and Darrell Laham (Jan. 1, 1998). “An Introduction to Latent Semantic Analysis”. In: *Discourse Processes* 25.2-3, pp. 259–284. ISSN: 0163-853X. DOI: 10.1080/01638539809545028. URL: <https://doi.org/10.1080/01638539809545028> (visited on 03/29/2024).
- Lazer, David et al. (Feb. 6, 2009). “Computational Social Science”. In: *Science* 323.5915, pp. 721–723. DOI: 10.1126/science.1167742. URL: <https://www.science.org/doi/full/10.1126/science.1167742> (visited on 06/15/2023).
- Lee, Daniel D. and H. Sebastian Seung (Oct. 1999). “Learning the Parts of Objects by Non-Negative Matrix Factorization”. In: *Nature* 401.6755, pp. 788–791. ISSN: 1476-4687. DOI: 10.1038/44565. URL: <https://www.nature.com/articles/44565> (visited on 03/29/2024).
- Levy, Omer and Yoav Goldberg (2014). “Neural Word Embedding as Implicit Matrix Factorization”. In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc. URL: <https://proceedings.neurips.cc/paper/2014/hash/feab05aa91085b7a8012516bc3533958-Abstract.html> (visited on 04/04/2024).
- Li, Yang and Tao Yang (2018). “Word Embedding for Understanding Natural Language: A Survey”. In: *Guide to Big Data Applications*. Ed. by S. Srinivasan. Studies in Big Data. Cham: Springer International Publishing, pp. 83–104. ISBN: 978-3-319-53817-4. DOI: 10.1007/978-3-319-53817-4_4. URL: https://doi.org/10.1007/978-3-319-53817-4_4 (visited on 07/26/2023).
- Liu, Qi, Matt J. Kusner, and Phil Blunsom (Apr. 13, 2020). *A Survey on Contextual Embeddings*. DOI: 10.48550/arXiv.2003.07278. arXiv: 2003.07278 [cs]. URL: <http://arxiv.org/abs/2003.07278> (visited on 07/31/2023). preprint.
- Lopez-Gazpio, I. et al. (Oct. 15, 2019). “Word N-Gram Attention Models for Sentence Similarity and Inference”. In: *Expert Systems with Applications* 132, pp. 1–11. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2019.04.054. URL: <https://www.sciencedirect.com/science/article/pii/S0957417419302842> (visited on 04/01/2024).
- Lund, Kevin and Curt Burgess (1996). “Producing High-Dimensional Semantic Spaces from Lexical Co-Occurrence”. In: *Behavior Research Methods, Instruments & Computers* 28.2, pp. 203–208. ISSN: 0743-3808. DOI: 10.3758/BF03204766.
- Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (July 7, 2008). *Introduction to Information Retrieval*. Higher Education from Cambridge University Press. DOI: 10.1017/CB09780511809071. URL: <https://www.cambridge.org/highereducation/books/introduction-to-information-retrieval/669D108D20F556C5C30957D63B5AB65C> (visited on 03/06/2024).
- Menezes, Telmo and Camille Roth (Feb. 18, 2021). *Semantic Hypergraphs*. DOI: 10.48550/arXiv.1908.10784. arXiv: 1908.10784 [cs]. URL: <http://arxiv.org/abs/1908.10784> (visited on 07/19/2022). preprint.

- Mihalcea, Rada, Courtney Corley, and Carlo Strapparava (July 16, 2006). “Corpus-Based and Knowledge-Based Measures of Text Semantic Similarity”. In: *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*. AAAI’06. Boston, Massachusetts: AAAI Press, pp. 775–780. ISBN: 978-1-57735-281-5.
- Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (June 2013). “Linguistic Regularities in Continuous Space Word Representations”. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL-HLT 2013. Atlanta, Georgia: Association for Computational Linguistics, pp. 746–751. URL: <https://aclanthology.org/N13-1090> (visited on 07/31/2023).
- Mikolov, Tomas et al. (Oct. 16, 2013a). *Distributed Representations of Words and Phrases and Their Compositionality*. DOI: 10.48550/arXiv.1310.4546. arXiv: 1310.4546 [cs, stat]. URL: <http://arxiv.org/abs/1310.4546> (visited on 06/22/2023). preprint.
- Mikolov, Tomas et al. (Sept. 6, 2013b). *Efficient Estimation of Word Representations in Vector Space*. DOI: 10.48550/arXiv.1301.3781. arXiv: 1301.3781 [cs]. URL: <http://arxiv.org/abs/1301.3781> (visited on 07/24/2023). preprint.
- Miller, George A. (Nov. 1, 1995). “WordNet: A Lexical Database for English”. In: *Communications of the ACM* 38.11, pp. 39–41. ISSN: 0001-0782. DOI: 10.1145/219717.219748. URL: <https://dl.acm.org/doi/10.1145/219717.219748> (visited on 03/27/2024).
- Min, Bonan et al. (Sept. 14, 2023). “Recent Advances in Natural Language Processing via Large Pre-trained Language Models: A Survey”. In: *ACM Computing Surveys* 56.2, 30:1–30:40. ISSN: 0360-0300. DOI: 10.1145/3605943. URL: <https://dl.acm.org/doi/10.1145/3605943> (visited on 03/05/2024).
- Mohammad, Saif M. and Graeme Hirst (Mar. 8, 2012a). *Distributional Measures as Proxies for Semantic Relatedness*. DOI: 10.48550/arXiv.1203.1889. arXiv: 1203.1889 [cs]. URL: <http://arxiv.org/abs/1203.1889> (visited on 03/28/2024). preprint.
- (Mar. 8, 2012b). *Distributional Measures of Semantic Distance: A Survey*. DOI: 10.48550/arXiv.1203.1858. arXiv: 1203.1858 [cs]. URL: <http://arxiv.org/abs/1203.1858> (visited on 03/26/2024). preprint.
- (Mar. 8, 2012c). *Distributional Measures of Semantic Distance: A Survey*. DOI: 10.48550/arXiv.1203.1858. arXiv: 1203.1858 [cs]. URL: <http://arxiv.org/abs/1203.1858> (visited on 03/27/2024). preprint.
- Niwa, Yoshiki and Yoshihiko Nitta (Aug. 1994). “Co-Occurrence Vectors From Corpora vs. Distance Vectors From Dictionaries”. In: *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*. COLING 1994. Kyoto, Japan. URL: <https://aclanthology.org/C94-1049> (visited on 03/29/2024).
- P., Sunilkumar and Athira P. Shaji (Dec. 2019). “A Survey on Semantic Similarity”. In: *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)*. 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), pp. 1–8. DOI: 10.1109/ICAC347590.2019.9036843.
- Parsons, Van L. (2017). “Stratified Sampling”. In: *Wiley StatsRef: Statistics Reference Online*. John Wiley & Sons, Ltd, pp. 1–11. ISBN: 978-1-118-44511-2. DOI: 10.1002/9781118445112.stat05999.pub2. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118445112.stat05999.pub2> (visited on 01/11/2024).
- Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. EMNLP 2014. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: <https://aclanthology.org/D14-1162> (visited on 07/26/2023).

- Qiu, XiPeng et al. (Oct. 1, 2020). “Pre-Trained Models for Natural Language Processing: A Survey”. In: *Science China Technological Sciences* 63.10, pp. 1872–1897. ISSN: 1869-1900. DOI: 10.1007/s11431-020-1647-3. URL: <https://doi.org/10.1007/s11431-020-1647-3> (visited on 06/15/2023).
- Rohde, Douglas LT, Laura M Gonnerman, and David C Plaut (2006). “An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence”. In: *Communications of the ACM* 8.627-633, p. 116.
- Rudin, Cynthia (May 2019). “Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead”. In: *Nature Machine Intelligence* 1.5 (5), pp. 206–215. ISSN: 2522-5839. DOI: 10.1038/s42256-019-0048-x. URL: <https://www.nature.com/articles/s42256-019-0048-x> (visited on 06/17/2023).
- Salle, Alexandre, Marco Idiart, and Aline Villavicencio (June 3, 2016). *Enhancing the LexVec Distributed Word Representation Model Using Positional Contexts and External Memory*. DOI: 10.48550/arXiv.1606.01283. arXiv: 1606.01283 [cs]. URL: <http://arxiv.org/abs/1606.01283> (visited on 07/31/2023). preprint.
- Salle, Alexandre and Aline Villavicencio (June 2018). “Incorporating Subword Information into Matrix Factorization Word Embeddings”. In: *Proceedings of the Second Workshop on Subword/Character LEvel Models*. SCLeM 2018. New Orleans: Association for Computational Linguistics, pp. 66–71. DOI: 10.18653/v1/W18-1209. URL: <https://aclanthology.org/W18-1209> (visited on 07/31/2023).
- (Aug. 19, 2019). *Why So Down? The Role of Negative (and Positive) Pointwise Mutual Information in Distributional Semantics*. DOI: 10.48550/arXiv.1908.06941. arXiv: 1908.06941 [cs]. URL: <http://arxiv.org/abs/1908.06941> (visited on 07/31/2023). preprint.
- Salle, Alexandre, Aline Villavicencio, and Marco Idiart (Aug. 2016). “Matrix Factorization Using Window Sampling and Negative Sampling for Improved Word Representations”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. ACL 2016. Berlin, Germany: Association for Computational Linguistics, pp. 419–424. DOI: 10.18653/v1/P16-2068. URL: <https://aclanthology.org/P16-2068> (visited on 07/31/2023).
- Sezerer, Erhan and Selma Tekir (Oct. 4, 2021). *A Survey On Neural Word Embeddings*. DOI: 10.48550/arXiv.2110.01804. arXiv: 2110.01804 [cs]. URL: <http://arxiv.org/abs/2110.01804> (visited on 07/26/2023). preprint.
- Shao, Yang (Aug. 2017). “HCTI at SemEval-2017 Task 1: Use Convolutional Neural Network to Evaluate Semantic Textual Similarity”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. SemEval 2017. Ed. by Steven Bethard et al. Vancouver, Canada: Association for Computational Linguistics, pp. 130–133. DOI: 10.18653/v1/S17-2016. URL: <https://aclanthology.org/S17-2016> (visited on 04/01/2024).
- Speer, Robyn, Joshua Chin, and Catherine Havasi (Feb. 12, 2017). “ConceptNet 5.5: An Open Multilingual Graph of General Knowledge”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 31.1 (1). ISSN: 2374-3468. DOI: 10.1609/aaai.v31i1.11164. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/11164> (visited on 03/27/2024).
- (Dec. 11, 2018). *ConceptNet 5.5: An Open Multilingual Graph of General Knowledge*. DOI: 10.48550/arXiv.1612.03975. arXiv: 1612.03975 [cs]. URL: <http://arxiv.org/abs/1612.03975> (visited on 07/31/2023). preprint.
- Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum (May 8, 2007). “Yago: A Core of Semantic Knowledge”. In: *Proceedings of the 16th International Conference on World*

- Wide Web*. WWW '07. New York, NY, USA: Association for Computing Machinery, pp. 697–706. ISBN: 978-1-59593-654-7. DOI: 10.1145/1242572.1242667. URL: <https://dl.acm.org/doi/10.1145/1242572.1242667> (visited on 03/27/2024).
- Sánchez, David and Montserrat Batet (Mar. 1, 2013). “A Semantic Similarity Method Based on Information Content Exploiting Multiple Ontologies”. In: *Expert Systems with Applications* 40.4, pp. 1393–1399. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2012.08.049. URL: <https://www.sciencedirect.com/science/article/pii/S095741741201010X> (visited on 03/27/2024).
- Tai, Kai Sheng, Richard Socher, and Christopher D. Manning (May 30, 2015). *Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks*. DOI: 10.48550/arXiv.1503.00075. arXiv: 1503.00075 [cs]. URL: <http://arxiv.org/abs/1503.00075> (visited on 04/01/2024). preprint.
- Tian, Junfeng et al. (Aug. 2017). “ECNU at SemEval-2017 Task 1: Leverage Kernel-based Traditional NLP Features and Neural Networks to Build a Universal Model for Multilingual and Cross-lingual Semantic Textual Similarity”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. SemEval 2017. Ed. by Steven Bethard et al. Vancouver, Canada: Association for Computational Linguistics, pp. 191–197. DOI: 10.18653/v1/S17-2028. URL: <https://aclanthology.org/S17-2028> (visited on 04/01/2024).
- Tien, Nguyen Huy et al. (Nov. 1, 2019a). “Sentence Modeling via Multiple Word Embeddings and Multi-Level Comparison for Semantic Textual Similarity”. In: *Information Processing & Management* 56.6, p. 102090. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2019.102090. URL: <https://www.sciencedirect.com/science/article/pii/S0306457319301335> (visited on 04/01/2024).
- (Nov. 1, 2019b). “Sentence Modeling via Multiple Word Embeddings and Multi-Level Comparison for Semantic Textual Similarity”. In: *Information Processing & Management* 56.6, p. 102090. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2019.102090. URL: <https://www.sciencedirect.com/science/article/pii/S0306457319301335> (visited on 04/01/2024).
- Turney, P. D. and P. Pantel (Feb. 27, 2010). “From Frequency to Meaning: Vector Space Models of Semantics”. In: *Journal of Artificial Intelligence Research* 37, pp. 141–188. ISSN: 1076-9757. DOI: 10.1613/jair.2934. URL: <https://www.jair.org/index.php/jair/article/view/10640> (visited on 03/28/2024).
- Vrandečić, Denny and Markus Krötzsch (Sept. 23, 2014). “Wikidata: A Free Collaborative Knowledgebase”. In: *Communications of the ACM* 57.10, pp. 78–85. ISSN: 0001-0782. DOI: 10.1145/2629489. URL: <https://dl.acm.org/doi/10.1145/2629489> (visited on 03/27/2024).
- Wang, Zhiguo, Haitao Mi, and Abraham Ittycheriah (July 14, 2017). *Sentence Similarity Learning by Lexical Decomposition and Composition*. DOI: 10.48550/arXiv.1602.07019. arXiv: 1602.07019 [cs]. URL: <http://arxiv.org/abs/1602.07019> (visited on 04/01/2024). preprint.
- Wieting, John et al. (June 1, 2015). “From Paraphrase Database to Compositional Paraphrase Model and Back”. In: *Transactions of the Association for Computational Linguistics* 3, pp. 345–358. ISSN: 2307-387X. DOI: 10.1162/tac1_a_00143. URL: https://doi.org/10.1162/tac1_a_00143 (visited on 04/04/2024).
- Wilkerson, John and Andreu Casas (May 1, 2017). *Large-Scale Computerized Text Analysis in Political Science: Opportunities and Challenges*. DOI: 10.1146/annurev-polisci-052615-025542. URL: <https://papers.ssrn.com/abstract=2968080> (visited on 06/15/2023). preprint.

- Young, Tom et al. (Nov. 24, 2018). *Recent Trends in Deep Learning Based Natural Language Processing*. DOI: 10.48550/arXiv.1708.02709. arXiv: 1708.02709 [cs]. URL: <http://arxiv.org/abs/1708.02709> (visited on 06/15/2023). preprint.
- Zad, Samira et al. (Oct. 2021). “A Survey of Deep Learning Methods on Semantic Similarity and Sentence Modeling”. In: *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), pp. 0466–0472. DOI: 10.1109/IEMCON53756.2021.9623078.
- Zheng, Tao et al. (Aug. 7, 2019). “Detection of Medical Text Semantic Similarity Based on Convolutional Neural Network”. In: *BMC Medical Informatics and Decision Making* 19.1, p. 156. ISSN: 1472-6947. DOI: 10.1186/s12911-019-0880-2. URL: <https://doi.org/10.1186/s12911-019-0880-2> (visited on 04/01/2024).
- Zhu, Ganggao and Carlos A. Iglesias (Jan. 2017). “Computing Semantic Similarity of Concepts in Knowledge Graphs”. In: *IEEE Transactions on Knowledge and Data Engineering* 29.1, pp. 72–85. ISSN: 1558-2191. DOI: 10.1109/TKDE.2016.2610428. URL: <https://ieeexplore.ieee.org/abstract/document/7572993> (visited on 03/27/2024).
- Zuccon, Guido et al. (Dec. 8, 2015). “Integrating and Evaluating Neural Word Embeddings in Information Retrieval”. In: *Proceedings of the 20th Australasian Document Computing Symposium*. ADCS '15. New York, NY, USA: Association for Computing Machinery, pp. 1–8. ISBN: 978-1-4503-4040-3. DOI: 10.1145/2838931.2838936. URL: <https://dl.acm.org/doi/10.1145/2838931.2838936> (visited on 04/04/2024).

A. Appendix

A.1. Reference Edge Sets

Num. of Ref. Edges	Ref. Edges Set ID	Reference Edge Content
1	1-1	Israeli gunfire wounds Gaza fisherman: ministry
1	1-2	Ukraine's Opposition Accuses Government of Provoking Violence
1	1-3	Thursday's attack by armed youths on the base in Bor left at least 58 dead, including children
1	1-4	Turkey suspends 15,200 education staff
1	1-5	Kurdish protesters storm the Conservative Party's campaign headquarters in London
3	3-1	Israeli gunfire wounds Gaza fisherman: ministry Kuwait Rejects Saudi Request for War Subvention Researchers Accuse Canadian Internet Company of Helping Yemen Censor the Web
3	3-2	Ukraine's Opposition Accuses Government of Provoking Violence Chinese island-building in the South China Sea is causing "irreversible and widespread damage to biodiversity and ecological balance," according to the Philippines; Manila accused China of disregarding the people who rely on the sea by destroying coral reefs to create new islands The government's opposition and various refugee organizations have harshly criticized the reforms
3	3-3	Thursday's attack by armed youths on the base in Bor left at least 58 dead, including children Venezuela expels 3 US consular officials Russia, China nix U.S. human rights claims: Russia and China disputed U.S. Ambassador Nikki Haley's contention that human rights violations are a main driver of conflicts
3	3-4	Turkey suspends 15,200 education staff Leading Muslim groups condemn ISIS killing of US journalists PayPal freezes Canadian media company's account over story about Syrian family
3	3-5	Kurdish protesters storm the Conservative Party's campaign headquarters in London Obama seeks new Syria strategy review to deal with ISIS Tougher Canadian visa policy hits foreign workers, protects Canadian jobs

Table A.1.: Edge content for the reference edge sets of size $N \in \{1, 3\}$

Num. of Ref. Edges	Ref. Edges Set ID	Reference Edge Content
10	10-1	<p>Israeli gunfire wounds Gaza fisherman: ministry</p> <p>Kuwait Rejects Saudi Request for War Subvention</p> <p>Researchers Accuse Canadian Internet Company of Helping Yemen Censor the Web</p> <p>Afghanistan President Ashraf Ghani slams Pakistan for harbouring terrorists, praises India</p> <p>5-year-old Kentucky boy fatally shoots 2-year-old sister</p> <p>Bangladesh police kill 'mastermind' of Dhaka cafe attack</p> <p>Philippines President Duterte orders army to destroy Islamic militants or risk ISIS disease</p> <p>Turkish jets kill 18 Daesh terrorists in northern Syria</p> <p>Report slams Israel's military law enforcement system</p> <p>Iran Pursuing Release of Sailors Abducted by Somalian Pirates</p>
10	10-2	<p>Ukraine's Opposition Accuses Government of Provoking Violence</p> <p>Chinese island-building in the South China Sea is causing "irreversible and widespread damage to biodiversity and ecological balance," according to the Philippines; Manila accused China of disregarding the people who rely on the sea by destroying coral reefs to create new islands</p> <p>The government's opposition and various refugee organizations have harshly criticized the reforms</p> <p>Syria and Russia oppose unilateral US strikes against ISIL in Syria</p> <p>Taliban attack in Afghanistan kills six policemen</p> <p>Turkish President condemns US commandos photographed sporting Kurdish militia insignia</p> <p>France could ease ban on gay men giving blood after ECJ ruling</p> <p>Libyan smuggler fighting kills 22 migrants</p> <p>Kazakhstan jails online editor for 'spreading false information'</p> <p>Russia urges Assad to give up chemical weapons</p>

Table A.2.: Edge content for the reference edge sets of size $N = 10$ (Part 1/3)

Num. of Ref. Edges	Ref. Edges Set ID	Reference Edge Content
10	10-3	<p>Thursday's attack by armed youths on the base in Bor left at least 58 dead, including children</p> <p>Venezuela expels 3 US consular officials</p> <p>Russia, China nix U.S. human rights claims: Russia and China disputed U.S. Ambassador Nikki Haley's contention that human rights violations are a main driver of conflicts</p> <p>Turkish PM tells female reporter to 'know your place</p> <p>South Korean prosecutors seek arrest of ex-President Park in corruption probe</p> <p>Taliban Announce Spring Offensive in Afghanistan</p> <p>Spain dismantles 'jihadist cell</p> <p>U.S. conducts 'counter terrorism strike' against al Qaeda-linked target in Libya</p> <p>Swiss prosecutors launch money-laundering probe against fugitive Ukrainian President Yanukovich, and son</p> <p>UK Wants 10 Year Prison Sentence For Online Pirates</p>
10	10-4	<p>Turkey suspends 15,200 education staff</p> <p>Leading Muslim groups condemn ISIS killing of US journalists</p> <p>PayPal freezes Canadian media company's account over story about Syrian family</p> <p>U.S. urges China's Xi to extend non-militarization pledge to all of South China Sea</p> <p>Sri Lanka accuses Canada of holding Commonwealth 'to ransom</p> <p>Russia and pro-Moscow rebels on Wednesday condemned Ukraine for ratifying two bills on greater autonomy for the separatist east, saying they violated a peace deal and threatened a shaky month-long truce</p> <p>Malaysia turns away 800 boat people; Thailand spots 3rd boat</p> <p>US expresses concern over security of Pakistan's Nuclear weapons</p> <p>Health experts accuse WHO of 'egregious failure' on Ebola</p> <p>Sunni militants accuse the army, perhaps the only widely respected public institution in Lebanon, of siding with Hezbollah</p>

Table A.3.: Edge content for the reference edge sets of size $N = 10$ (Part 2/3)

Num. of Ref. Edges	Ref. Edges Set ID	Reference Edge Content
10	10-5	<p>Kurdish protesters storm the Conservative Party's campaign headquarters in London</p> <p>Obama seeks new Syria strategy review to deal with ISIS</p> <p>Tougher Canadian visa policy hits foreign workers, protects Canadian jobs</p> <p>U.S. preparing new sanctions against Chinese entities over financial support to North Korea</p> <p>Turkey's Erdogan makes Nazi jibe over Germany rally ban: "Your practices are not different from the Nazi practices of the past"</p> <p>Brazil committee recommends Dilma Rousseff's impeachment</p> <p>U.S. dismisses Russian concern about missile defense system in South Korea</p> <p>South Korea mulls ban on bosses messaging employees at home</p> <p>Israel to destroy homes of Palestinian Jerusalem car attackers</p> <p>Majority of Finns reject NATO membership</p>

Table A.4.: Edge content for the reference edge sets of size $N = 10$ (Part 3/3)

A.2. Best F1-score based Eval. Run Comparison Tables

Evaluation Run Name				Prec.	Rec.	(Best) F1-Score	
CP	NM	RES	t_s			Std.	Dev.
semsim-ctx	e5	r-1-1	0.65	0.386	0.768	0.514	-
semsim-ctx	e5	r-1-2	0.67	0.442	0.769	0.562	-
semsim-ctx	e5	r-1-3	0.59	0.372	0.838	0.515	-
semsim-ctx	e5	r-1-4	0.67	0.363	0.804	0.500	-
semsim-ctx	e5	r-1-5	0.68	0.397	0.681	0.502	-
semsim-ctx	e5	r-1-mean	0.64	0.356	0.823	0.477	+/- 0.046
semsim-ctx	e5	r-1-mean-best	0.65	0.392	0.772	0.518	+/- 0.025
semsim-ctx	e5	r-3-1	0.68	0.398	0.836	0.539	-
semsim-ctx	e5	r-3-2	0.68	0.435	0.752	0.551	-
semsim-ctx	e5	r-3-3	0.69	0.410	0.846	0.553	-
semsim-ctx	e5	r-3-4	0.69	0.392	0.846	0.536	-
semsim-ctx	e5	r-3-5	0.68	0.361	0.814	0.500	-
semsim-ctx	e5	r-3-mean	0.69	0.417	0.753	0.534	+/- 0.024
semsim-ctx	e5	r-3-mean-best	0.68	0.399	0.818	0.536	+/- 0.021
semsim-ctx	e5	r-10-1	0.71	0.415	0.817	0.550	-
semsim-ctx	e5	r-10-2	0.72	0.435	0.793	0.562	-
semsim-ctx	e5	r-10-3	0.72	0.400	0.771	0.527	-
semsim-ctx	e5	r-10-4	0.72	0.454	0.735	0.562	-
semsim-ctx	e5	r-10-5	0.71	0.378	0.835	0.520	-
semsim-ctx	e5	r-10-mean	0.72	0.428	0.746	0.543	+/- 0.021
semsim-ctx	e5	r-10-mean-best	0.72	0.416	0.790	0.544	+/- 0.020
semsim-ctx	gte	r-1-1	0.60	0.349	0.794	0.485	-
semsim-ctx	gte	r-1-2	0.63	0.388	0.799	0.522	-
semsim-ctx	gte	r-1-3	0.57	0.303	0.972	0.462	-
semsim-ctx	gte	r-1-4	0.55	0.300	1.000	0.461	-
semsim-ctx	gte	r-1-5	0.61	0.339	0.829	0.482	-
semsim-ctx	gte	r-1-mean	0.60	0.327	0.870	0.474	+/- 0.013
semsim-ctx	gte	r-1-mean-best	0.59	0.336	0.879	0.483	+/- 0.025
semsim-ctx	gte	r-3-1	0.64	0.364	0.826	0.505	-
semsim-ctx	gte	r-3-2	0.67	0.378	0.713	0.494	-
semsim-ctx	gte	r-3-3	0.67	0.385	0.740	0.506	-
semsim-ctx	gte	r-3-4	0.66	0.376	0.820	0.516	-
semsim-ctx	gte	r-3-5	0.62	0.322	0.896	0.474	-
semsim-ctx	gte	r-3-mean	0.66	0.374	0.714	0.486	+/- 0.040
semsim-ctx	gte	r-3-mean-best	0.65	0.365	0.799	0.499	+/- 0.016
semsim-ctx	gte	r-10-1	0.67	0.377	0.869	0.526	-
semsim-ctx	gte	r-10-2	0.67	0.361	0.917	0.518	-
semsim-ctx	gte	r-10-3	0.69	0.382	0.834	0.524	-
semsim-ctx	gte	r-10-4	0.69	0.399	0.730	0.516	-
semsim-ctx	gte	r-10-5	0.68	0.388	0.708	0.501	-
semsim-ctx	gte	r-10-mean	0.68	0.378	0.803	0.513	+/- 0.009
semsim-ctx	gte	r-10-mean-best	0.68	0.382	0.812	0.517	+/- 0.010

Table A.5.: Best F1-score evaluation runs for all ERS of the form `semsim-ctx NM r-N-X` (CNESS with AT option disabled)

Evaluation Run Name				Prec.	Rec.	(Best) F1-Score	
CP	NM	RES	t_s				Std. Dev.
semsim-ctx	e5-at	r-1-1	0.68	0.338	0.915	0.494	-
semsim-ctx	e5-at	r-1-2	0.71	0.434	0.701	0.536	-
semsim-ctx	e5-at	r-1-3	0.66	0.356	0.885	0.508	-
semsim-ctx	e5-at	r-1-4	0.71	0.365	0.798	0.501	-
semsim-ctx	e5-at	r-1-5	0.71	0.351	0.908	0.507	-
semsim-ctx	e5-at	r-1-mean	0.69	0.350	0.847	0.487	+/- 0.021
semsim-ctx	e5-at	r-1-mean-best	0.69	0.369	0.841	0.509	+/- 0.016
semsim-ctx	e5-at	r-3-1	0.72	0.363	0.846	0.508	-
semsim-ctx	e5-at	r-3-2	0.72	0.389	0.765	0.516	-
semsim-ctx	e5-at	r-3-3	0.73	0.399	0.780	0.528	-
semsim-ctx	e5-at	r-3-4	0.73	0.386	0.829	0.526	-
semsim-ctx	e5-at	r-3-5	0.72	0.351	0.886	0.503	-
semsim-ctx	e5-at	r-3-mean	0.73	0.392	0.740	0.511	+/- 0.016
semsim-ctx	e5-at	r-3-mean-best	0.72	0.378	0.821	0.516	+/- 0.011
semsim-ctx	e5-at	r-10-1	0.74	0.382	0.849	0.527	-
semsim-ctx	e5-at	r-10-2	0.75	0.395	0.781	0.525	-
semsim-ctx	e5-at	r-10-3	0.75	0.405	0.815	0.541	-
semsim-ctx	e5-at	r-10-4	0.74	0.371	0.890	0.523	-
semsim-ctx	e5-at	r-10-5	0.74	0.357	0.881	0.509	-
semsim-ctx	e5-at	r-10-mean	0.75	0.395	0.766	0.521	+/- 0.016
semsim-ctx	e5-at	r-10-mean-best	0.74	0.382	0.843	0.525	+/- 0.012
semsim-ctx	gte-at	r-1-1	0.65	0.342	0.826	0.483	-
semsim-ctx	gte-at	r-1-2	0.68	0.386	0.774	0.515	-
semsim-ctx	gte-at	r-1-3	0.66	0.311	0.945	0.468	-
semsim-ctx	gte-at	r-1-4	0.63	0.300	1.000	0.461	-
semsim-ctx	gte-at	r-1-5	0.66	0.339	0.866	0.487	-
semsim-ctx	gte-at	r-1-mean	0.66	0.328	0.882	0.476	+/- 0.018
semsim-ctx	gte-at	r-1-mean-best	0.66	0.335	0.882	0.483	+/- 0.021
semsim-ctx	gte-at	r-3-1	0.69	0.340	0.827	0.482	-
semsim-ctx	gte-at	r-3-2	0.69	0.309	0.950	0.467	-
semsim-ctx	gte-at	r-3-3	0.72	0.348	0.852	0.494	-
semsim-ctx	gte-at	r-3-4	0.71	0.367	0.822	0.508	-
semsim-ctx	gte-at	r-3-5	0.67	0.317	0.930	0.473	-
semsim-ctx	gte-at	r-3-mean	0.69	0.324	0.883	0.472	+/- 0.013
semsim-ctx	gte-at	r-3-mean-best	0.70	0.336	0.876	0.485	+/- 0.017
semsim-ctx	gte-at	r-10-1	0.71	0.342	0.879	0.492	-
semsim-ctx	gte-at	r-10-2	0.72	0.336	0.910	0.491	-
semsim-ctx	gte-at	r-10-3	0.73	0.349	0.891	0.501	-
semsim-ctx	gte-at	r-10-4	0.72	0.341	0.885	0.492	-
semsim-ctx	gte-at	r-10-5	0.70	0.322	0.934	0.478	-
semsim-ctx	gte-at	r-10-mean	0.72	0.341	0.854	0.486	+/- 0.002
semsim-ctx	gte-at	r-10-mean-best	0.72	0.338	0.900	0.491	+/- 0.008

Table A.6.: Best F1-score evaluation runs for all ERS of the form `semsim-ctx NM-at r-N-X` (CNESS with AT option enabled)

A.3. Evaluation Metric Scores vs. Similarity Threshold Plots

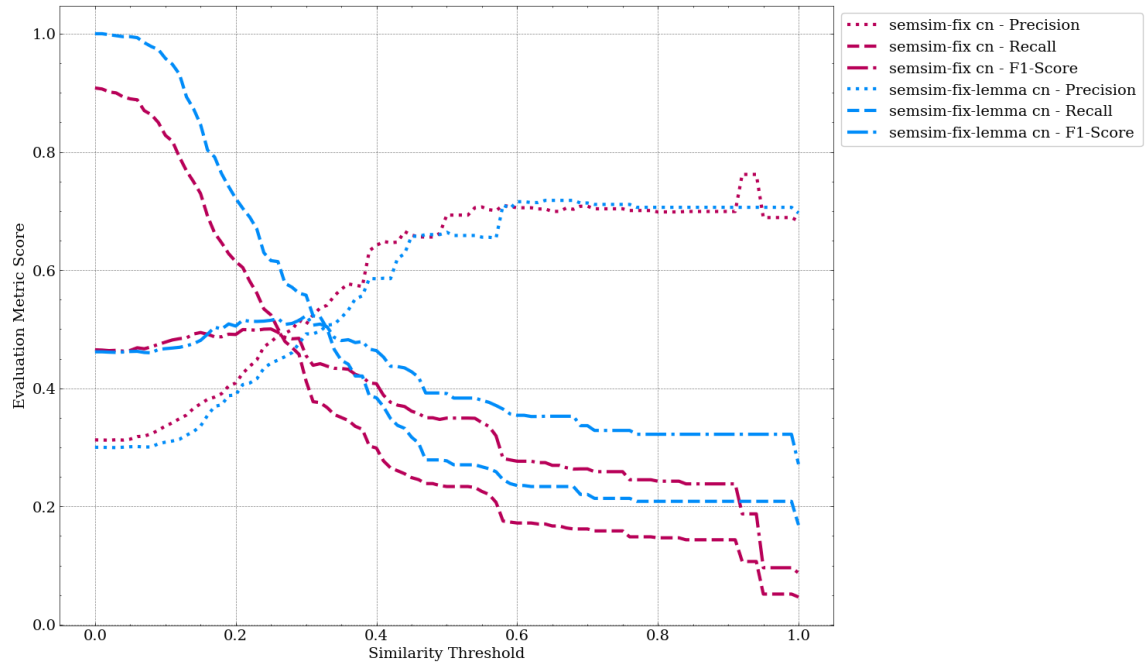


Figure A.1.: Precision, recall and F1-score vs. ST for the evaluation runs `semsim-fix cn` and `semsim-fix-lemma cn`

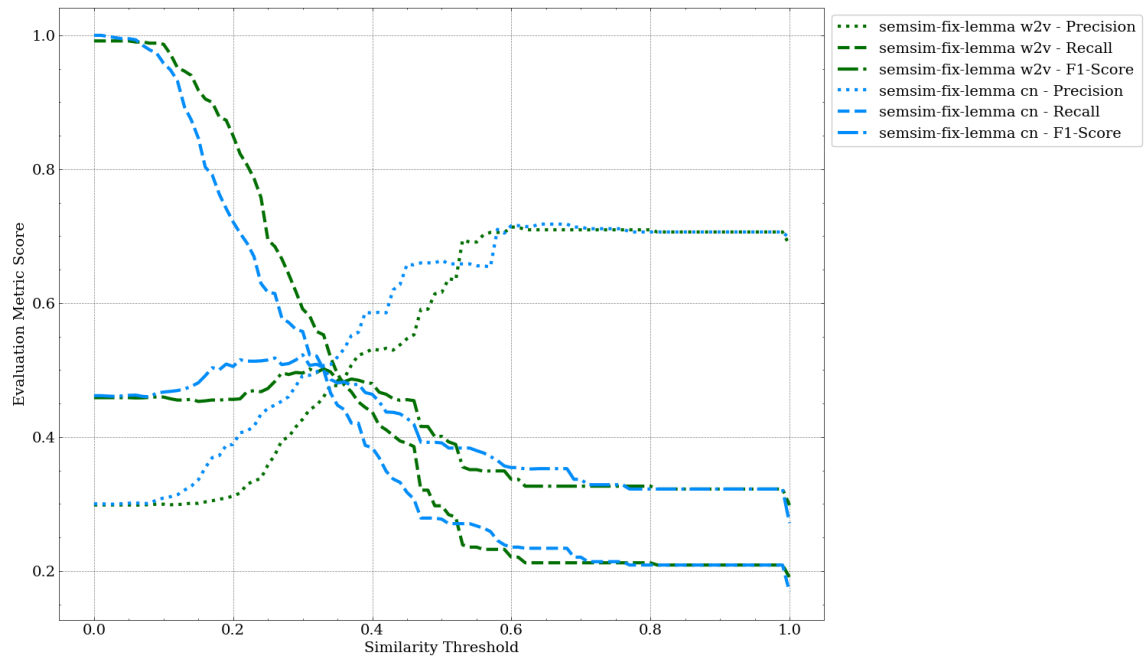


Figure A.2.: Precision, recall and F1-score vs. ST for the evaluation runs `semsim-fix-lemma w2v` and `semsim-fix-lemma cn`

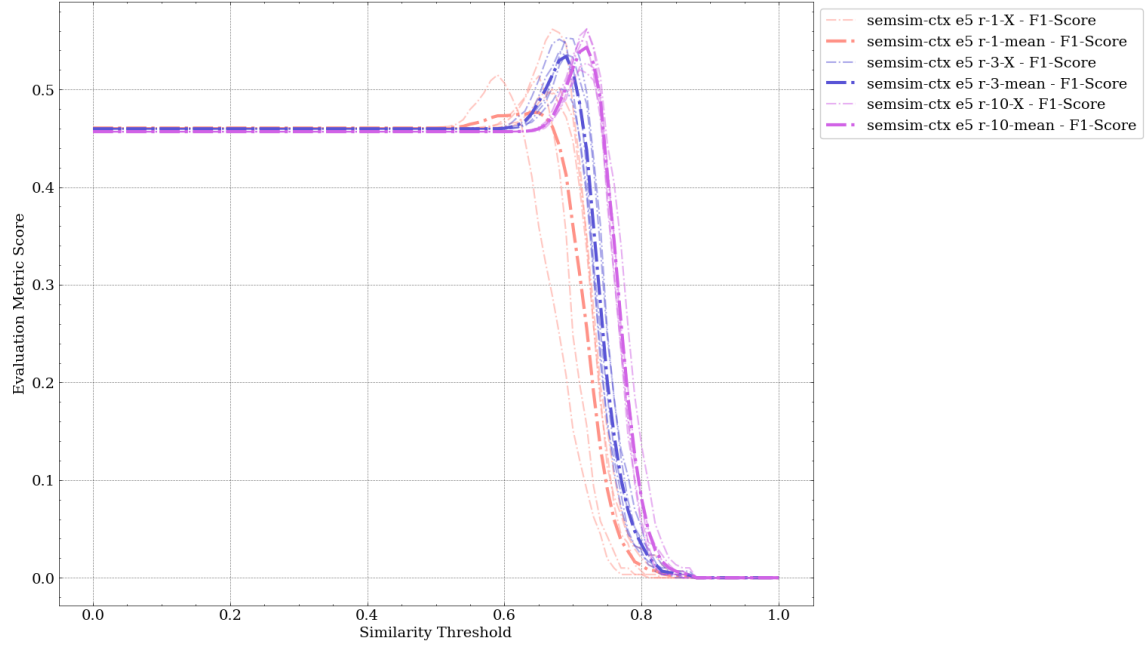


Figure A.3.: F1-score vs. ST for the evaluation runs `semsim-ctx e5 r-1-X`, `semsim-ctx e5 r-3-X` and `semsim-ctx e5 r-10-X`

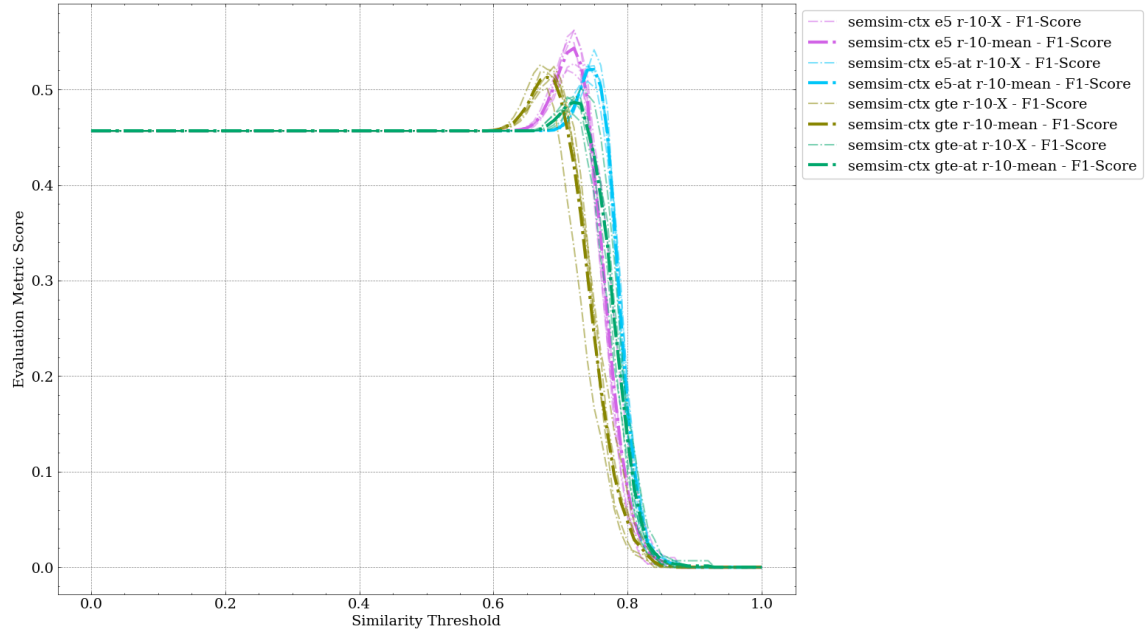


Figure A.4.: F1-score vs. ST for the evaluation runs `semsim-ctx e5 r-10-X`, `semsim-ctx e5-at r-10-X`, `semsim-ctx gte r-10-X` and `semsim-ctx gte-at r-10-X`

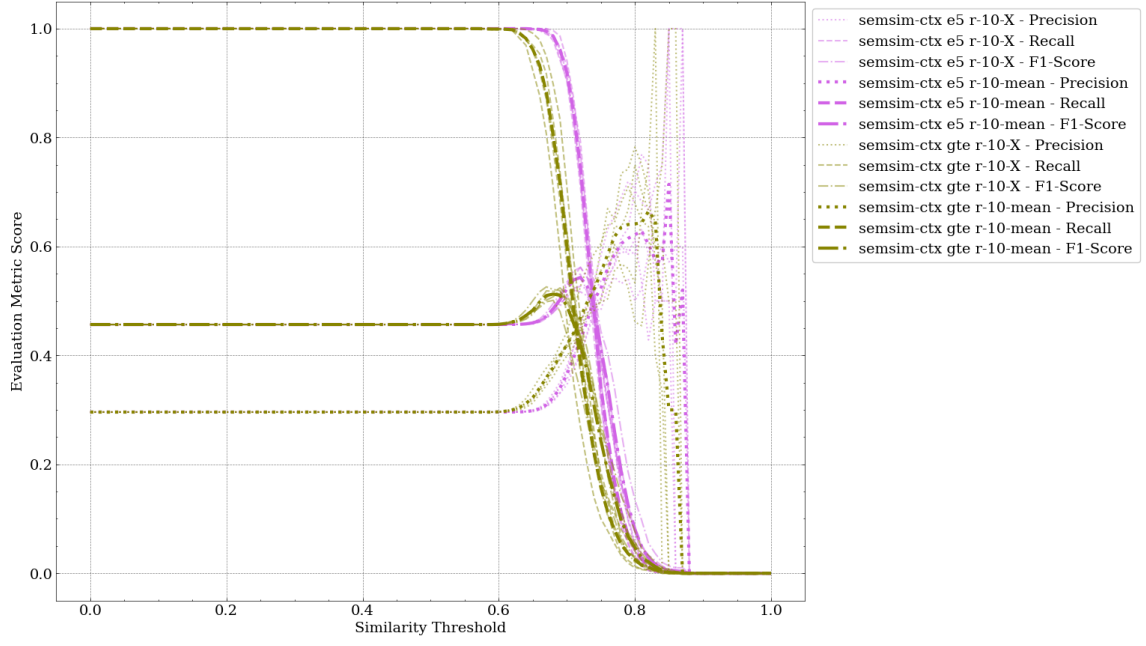


Figure A.5.: Precision, recall and F1-score vs. ST for the evaluation runs `semsim-ctx e5 r-10-X` and `semsim-ctx gte r-10-X`

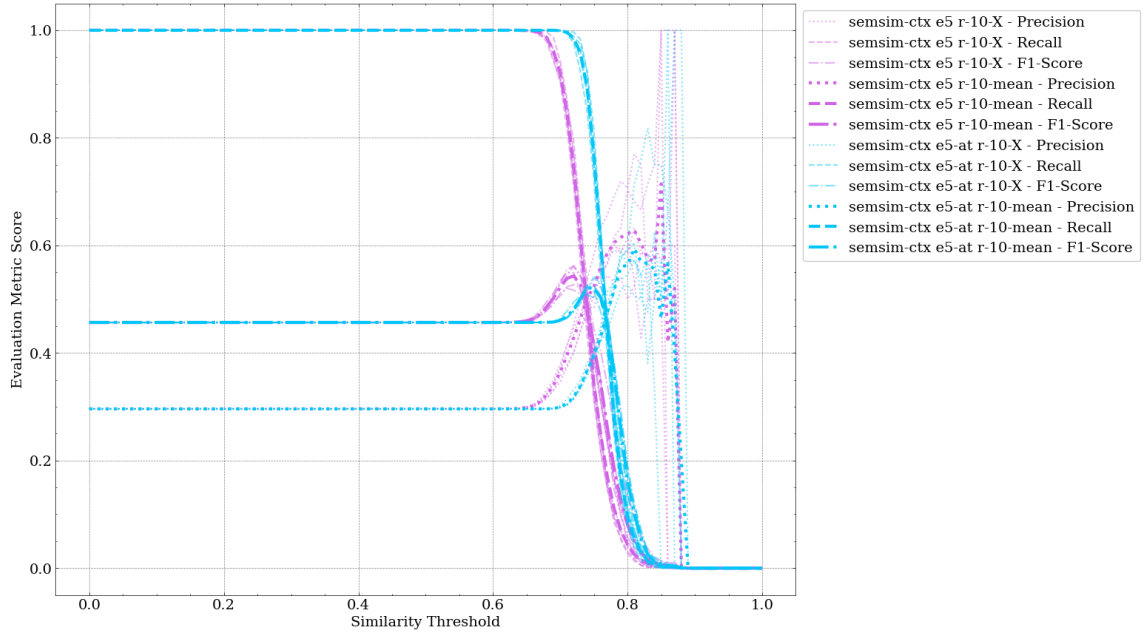


Figure A.6.: Precision, recall and F1-score vs. ST for the evaluation runs `semsim-ctx e5 r-10-X` and `semsim-ctx e5-at r-10-X`