Leveraging Approximate Pattern Matching with BERT for Event Detection

Hristo Tanev
European Commission, Joint Research Centre,
via Enrico Fermi 2749,
Ispra 21020, Italy
hristo.tanev@ec.europa.eu

Abstract

We describe a new weakly supervised method for sentence-level event detection, based exclusively on linear prototype patterns. We propose a BERT based algorithm for approximate pattern matching to identify event phrases, semantically similar to these prototypes. To the best of our knowledge, this is the first time a similar approach is used in the context of event detection. We experimented with two event corpora in the area of disease outbreaks and terrorism and we achieved promising results in sentence level event identification, 0.78 F1 score for new disease cases and 0.68 F1 for terrorist attacks. Results were in line with two state-of-the art systems, based on supervised ML and sophisticated linguistic rules.

1 Introduction

Early event extraction systems predominantly rely on pattern matching and linguistic rules (Xiang and Wang, 2019). This approach remains particularly effective in well-defined domains, such as disease outbreaks, biomedical papers, disasters, security, and socio-political developments, where language is clearly structured Tanev et al. (2008); Valenzuela-Escárcega et al. (2015); Nitschke et al. (2022).

In specific contexts, linguistic rules can offer competitive precision and enhanced transparency compared to machine learning (ML) models (Chiticariu et al., 2013). Linguistic rules can also be used for automatic corpus annotation, when new domains of event extraction are being considered and no training data is available (Wang et al., 2019). The transparency inherent in linguistic rules is particularly vital in real-world event extraction applications. End users can provide feedback on the performance of specific keywords and phrases, thereby improving the accuracy and breadth of the rule set.

In this work we argue that the combination of manually crafted linear patterns and Large Language Models (LLM) is a promising avenue for combining the strengths of the knowledge based approaches and the LLM in the domain of event detection. We experimented in the domains of security and health, but we think that the approach is applicable across a wide range of domains and event classes. LLM like BERT (Devlin et al., 2018) offer the capability to create utterance abstractions, using the contextualized word embeddings, which can be received from the embedding layer of the BERT neural network, see Figure 1. In the context of event detection, this allows for creating simple linear patterns as prototypes, e.g. "people have got a disease", and using the BERT contextualized embeddings of both prototypes and analysed text to find the semantic relation between the patterns and their lexical variations in the text, e.g. "children have got influenza".

More concretely, we propose the following approach, starting with prototype patterns (here we consider the event types *new disease cases* and *terrorist attacks*) like "disease outbreak", "number people were infected" or "bomb exploded", to discovers in the test set sentences containing text fragments, containing words with similar BERT embedding vectors - "influenza outbreak", "COVID was discovered in 2 foreign nationals", or "blast killed", etc. These phrases are supposed to be semantically similar to the prototypes, because of their similarity in the BERT encoding. Our experiments demonstrated that BERT-based pattern matching is able to infer event mentions which have significant lexical and syntactic differences with respect to the prototype patterns.

We tested our approach on the task of detecting sentences containing events of a predefined event type. Two event classes were considered in our experiments: *new disease cases* and *terrorist attacks*. For both of them we achieved performance much higher than the baseline. Moreover, for the event type *new disease cases*, the achieved performance was in line with other systems, based on supervised

Proceedings of the 7th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2024), pages 32–39
March 22, 2024 ©2024 Association for Computational Linguistics

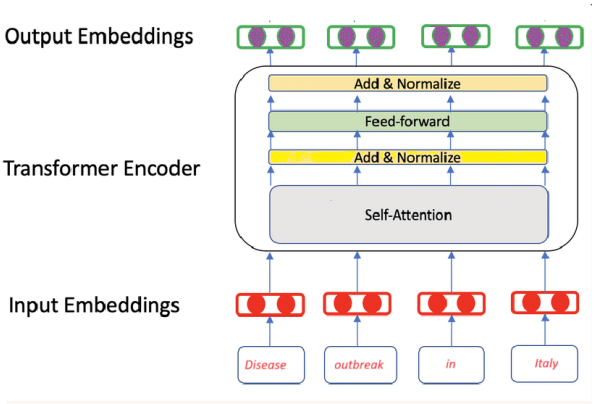

Figure 1: BERT contextualized embeddings

ML.

2 Related work

Event detection at various levels: token, sentence and document level has been largely addressed in previous work. The CASE shared task on protest detection and the participating systems (Hürriyetoğlu et al., 2021) tackle event detection at all the three levels. Various methods for sentence-level event detection are studied in Naughton et al. (2010).

A survey of the existing event detection and extraction approaches were presented in (Hogenboom et al., 2011) and (Xiang and Wang, 2019)

Pattern matching is a well established method for extracting event triggers and arguments. Earlier event extraction systems massively exploited lexico-syntactic patterns (Xiang and Wang, 2019). Most of these systems used domain specific grammars and ontologies in complex linguistic patterns. It is noteworthy that some of the state-of-the-art systems in the domain of security use linear patterns and linguistic rules Tanev et al. (2008); Atkinson et al. (2013); Nitschke et al. (2022). Similarly, rule-based event extraction is used in the biomedical domain Bui et al. (2013); Valenzuela-Escárcega et al. (2015).

Related to the prototype pattern matching we propose here, is another BERT based entity matching approach (Paganelli et al., 2022). BERT pattern matching is also used in Question Answering to find similar questions (Wang et al., 2020). Supervised learning, considering event triggers, is described in several works: (Liao et al., 2021), (Hao et al., 2023), (Lai et al., 2021), and (Tuo et al., 2023).

3 The approach

The approximate pattern matching approach is designed to identify in a test corpus the sentences containing event descriptions. At the same time, the algorithm identifies an n-gram in each of these sentences, matching best one of the prototypes.

In our experiments we considered two event types - *new disease cases* and *terrorist attacks*, however we think that the method is applicable across various domains and event classes.

3.1 User-Generated Pattern Set

The foundation of our method rests on an input set of prototypes, which are linear event detection patterns. The prototypes are phrases describing event triggers together with the most important event arguments, such as actors or victims. In this experiment, we used as triggers and arguments generic concepts, such as "disease", "people", "sick", etc. A sentence containing a phrase semantically similar to one of the prototype patterns should indicate the presence of the targeted event. In the context of disease outbreak detection, relevant patterns include "people got disease," "people are sick," "new cases of disease," and "disease outbreak," encapsulating generic concepts such as "people" and "disease."

To perform approximate pattern matching, we leverage BERT context embeddings (Figure 1), comparing the token embeddings of each pattern with the token embeddings of the test sentence. The vector sequence matching is described in the following subsection. The matching process enables the identification of texts containing more

Corpus	Event type	All sent.	Positive sent.	Patterns
Disease outbreaks	New disease cases	212	62	40
Political violence and disasters	Terrorist attack	994	97	21

Table 1: Test data and evaluation settings

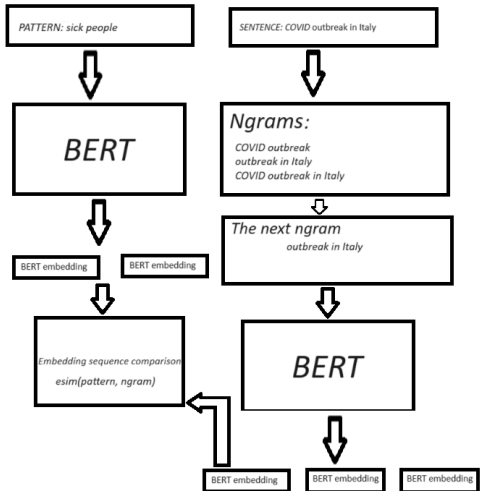

Figure 2: Comparing pattern and a sentence

concrete concepts or synonyms instantiating the generic ones from the prototypes, illustrated by event phrases like "students got influenza," "men are ill," "new cases of COVID," and "zika outbreak."

In our experimental configuration, we opted for the use of generic concepts, such as "people" in the input set of patterns, driven by the simplicity in pattern creation. Nevertheless, we have to acknowledge that the incorporation of concrete concepts, such as in "children got flu," is also a viable prototyping approach.

3.2 Calculating sentence eventness via approximate pattern matching

Given a set of event detection patterns

$Patterns = p_1, p_2, \ldots, p_n$ and a sentence s,

the approximate pattern recognition is used to calculate the *sentence eventness* of s, a non probability function in the interval $[0, 1]$, which shows how well the best pattern from the sequence matches the text.

Since patterns should be created in such a way that they describe unambiguously an event, the $eventness$ score should also be indicative about the likelihood of s containing an event of the specified class. Clearly, various discourse phenomena like questioning, conditional statements, negation, semantic ambiguity and others can play a role in preventing pattern matching from estimating correctly the sentence eventness.

Below, we outline the procedure for calculating the "eventness" of a sentence s. Steps 1 to 5 of this algorithm are also shown on Figure 2.

1. Each pattern $p_i = w_1 w_2 ... w_{in}$, where w_k is a word, generates a sequence of BERT embedding vectors, one for each word w_k. The sequence is denoted as $es(p_i)$.

2. From s, all word ngrams with size between 2 and 20 words are generated, denoted as $ngrams(s)$.

   For example, for the sentence s ="The crowd in Damascus shouted slogans.", $ngrams(s)$ ={"crowd in Damascus", "crowd in Damascus shouted", crowd in Damascus shouted slogans", "Damascus shouted", "Damascus shouted slogans", "shouted slogans"}

3. The sentence s is transformed into a sequence of word embedding vectors $es(s)$, in the same way $es(p_i)$ was obtained in step 1.

4. For each ngram $ng \in ngrams(s)$, its subsequence of corresponding embedding vectors is taken from $es(s)$. For example, for the ngram

Event type	$\alpha = 0.6$	$\alpha = 0.65$	$\alpha = 0.7$	Baseline
New disease cases	0.75	**0.78**	0.65	0.32
Terrorist attack	0.61	0.68	0.68	0.09
Macro average	0.68	0.73	0.67	0.21

Table 2: F1 score for various thresholds and baseline "exact pattern matching"

"crowd in Damascus", we will take the second, third and fourth embedding vector from $es(s)$. We denote the subsequence of embedding vectors for ng as $ses(ng, s)$.

5. Finally, we propose a similarity function $esim$ for comparing sequences of embedding vectors and calculating the eventness of s, $ev(s)$, via the following formula:
   $ev(s) = \max_{p,ng} esim(es(p), ses(ng, s))$, where $p \in Patterns, ng \in ngrams(s)$.

6. If $ev(s) > \alpha$, then s is considered a sentence containing an event. The threshold α is being set empirically.

We describe in details the eventness calculation in Appendix A and in Figure 3.

4 Experiments

To assess the efficacy and adaptability of our event detection methodology, we collected a test set of two distinct event corpora, each derived from a disparate domain: disease outbreaks (Piskorski et al., 2023) and politically motivated violence and disasters (Atkinson et al., 2017a). A unique targeted event type was specified for each corpus. Table 1 shows the parameters of the corpora and the targeted event types.

Our approach involves the systematic crafting of a set of carefully tailored linear patterns for the specific event types. The formulation of patterns drew upon insights derived from a development set, encompassing 300 sentences extracted from each respective corpus. This was done in the following steps:

1. We have created an initial set of patterns using our knowledge of the domain, getting additional insights from the development set.

2. Then, we matched these patterns on the sentences from the development corpus, using our approximate pattern matching algorithm.

3. We analyzed in random a subset of the false positives and false negatives.

4. We deleted patterns generating many false positives and created new patterns to detect the false negatives

5. This pattern development cycle was repeated several times (3-7) for each event type.

Generally, the creation of linguistic patterns is a intricate process, usually encompassing a combination of machine learning techniques and expert assessment (Tanev et al., 2009). However, in this study, our approach involved crafting patterns primarily based on linguistic expertise, with the development set used to assess their coverage and precision. The pattern development process was not the central focus of our work. Instead, we followed a pragmatic approach akin to what an average pattern developer might undertake, aiming for optimal results without substantial time investment.

In order to test the accuracy of the prototypes, we randomly selected a test subset from each corpus, non overlapping with the development set. Table 1 provides a comprehensive overview of the parameters defining each test set. These encompass the corpus name, targeted event type, total sentence count, and the frequency of positive instances (the sentences featuring the targeted event type), along with the number of developed linear patterns.

Before we run the algorithm, we had to set the α eventness threshold. Our observation on the development set was that the threshold delivers meaningful results in the interval 0.6 to 0.7. Therefore, we have run the evaluation with three different values for α: 0.6, 0.65, and 0.7.

We applied our approximate pattern matching detection of sentences on each corpus for each of the three α threshold values. Table 2 reports the obtained F1 score for each of the three thresholds.

We have also defined a baseline - *exact pattern matching*: if even one pattern is contained as a substring in a sentence, then the sentence is considered to contain an event. In Table 2 we report the F1 measure of this baseline.

Experiments showed that our method outpaced by a considerable margin the baseline. At the same

Matching n-gram with surroundings	Pattern
...the **number of confirmed** COVID-19 cases...	number of infected
The **number of Zika virus cases** has crossed 100 ...	number of infected
...raising the **death toll due to the disease** to 11 ...	death toll from the outbreak
...the situation where the **observed number of cases** exceeds...	number of infected
...the number of people **testing positive for the infection** rose...	testing positive for virus
...321 new domestically **transmitted coronavirus cases**...	confirmed disease cases
...**proportion of those testing** positive to the total tests...	number of infected
...**New clusters of coronavirus infections are igniting** concerns...	new infection cases
...new **confirmed coronavirus infections** have hit a record...	confirmed disease cases

Table 3: Patterns and their **matching n-grams** with the surrounding sentence fragment

time, the performance of the event class *new disease cases* achieved quite promising $F1$ score. Although conducted on different test sets, it is worth mentioning that this $F1$ score is in line with the accuracy achieved by some supervised systems in the outbreak detection domain. (Conway et al., 2009; Khatua et al., 2019).

Approximate pattern matching showed lower accuracy on the terrorist attacks with respect to the disease cases detection, still the $F1$ score stayed close to the performance of another early event extraction system in the area of security (Tanev et al., 2008; Atkinson et al., 2017b). It's important to emphasize that these evaluations were conducted on different corpora, providing only a general and imprecise basis for comparison.

Analysing the errors for the terrorist attack event type, we saw that there are text fragments matched against the patterns, where terrorists were victims, rather than attackers. Some sentences describing assassinations and kidnappings, especially in the Middle East were also erroneously labeled as terrorist attacks. For example, the phrase "victim of an assassination attempt" erroneously matched the pattern "victim of a terrorist attack". Also "air raid killed civilians" erroneously matched the pattern "market bomb targeted civilians". These and other pattern matching errors clearly show that in some cases the BERT pattern matching may be misled by particular phrases in certain contexts.

5 Conclusions

Results from Table 2 indicate that our approach attains satisfactory accuracy; nonetheless, its performance may vary across event classes. The performance, achieved in the detection of *new disease cases*, was a notable outcome considering the absence of supervision and the comparable accuracy observed in other supervised systems for detection of disease reports, (Conway et al., 2009), (Khatua et al., 2019). In Table 3 we show some of the prototypes for new disease cases and their matching n-grams in context. It is evident that approximate pattern matching can capture various syntactic and lexical variations.

Moreover, some of the detected n-grams are relevant as event detecting phrases and they themselves can constitute prototypes. Following this line of thinking, the approximate pattern matching algorithm can also be used for learning of new patterns.

As a conclusion, our experiments show that BERT-based pattern matching is an efficient weakly supervised event classifier. This method combines the simplicity and transparency of the pattern-based approaches and the implicit semantic knowledge, encoded in large language models like BERT.

References

Martin Atkinson, Mian Du, Jakub Piskorski, Hristo Tanev, Roman Yangarber, and Vanni Zavarella. 2013. Techniques for multilingual security-related event extraction from online news. *Computational Linguistics: Applications*, pages 163–186.

Martin Atkinson, Jakub Piskorski, Hristo Tanev, and Vanni Zavarella. 2017a. On the creation of a security-related event corpus. In *Proceedings of the Events and Stories in the News Workshop*, pages 59–65.

Martin Atkinson, Jakub Piskorski, Hristo Tanev, and Vanni Zavarella. 2017b. On the creation of a security-related event corpus. In *Proceedings of the Events and Stories in the News Workshop*, pages 59–65, Vancouver, Canada. Association for Computational Linguistics.

Quoc-Chinh Bui, David Campos, Erik van Mulligen, and Jan Kors. 2013. A fast rule-based approach for biomedical event extraction. In *proceedings of the BioNLP shared task 2013 workshop*, pages 104–108.

Laura Chiticariu, Yunyao Li, and Frederick Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 827–832.

Mike Conway, Son Doan, Ai Kawazoe, and Nigel Collier. 2009. Classifying disease outbreak reports using n-grams and semantic features. *International journal of medical informatics*, 78(12):e47–e58.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Anran Hao, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2023. A contrastive learning framework for event detection via semantic type prototype representation modelling. *Neurocomputing*, 556:126613.

Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, and Franciska De Jong. 2011. An overview of event extraction from text. *DeRiVE@ ISWC*, pages 48–57.

Ali Hürriyetoğlu, Hristo Tanev, Vanni Zavarella, Jakub Piskorski, Reyyan Yeniterzi, and Erdem Yörük. 2021. Challenges and applications of automated extraction of socio-political events from text (CASE 2021): Workshop and shared task report. *arXiv preprint arXiv:2108.07865*.

Aparup Khatua, Apalak Khatua, and Erik Cambria. 2019. A tale of two epidemics: Contextual word2vec for classifying twitter streams during outbreaks. *Information Processing & Management*, 56(1):247–257.

Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Learning prototype representations across few-shot tasks for event detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*.

Jinzhi Liao, Xiang Zhao, Xinyi Li, Lingling Zhang, and Jiuyang Tang. 2021. Learning discriminative neural representations for event detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 644–653.

Martina Naughton, Nicola Stokes, and Joe Carthy. 2010. Sentence-level event classification in unstructured texts. *Information retrieval*, 13:132–156.

Remo Nitschke, Yuwei Wang, Chen Chen, Adarsh Pyarelal, and Rebecca Sharp. 2022. Rule based event extraction for artificial social intelligence. In *Proceedings of the First Workshop on Pattern-based Approaches to NLP in the Age of Deep Learning*, pages 71–84, Gyeongju, Republic of Korea. International Conference on Computational Linguistics.

Matteo Paganelli, Francesco Del Buono, Andrea Baraldi, Francesco Guerra, et al. 2022. Analyzing how BERT performs entity matching. *Proceedings of the VLDB Endowment*, 15(8):1726–1738.

Jakub Piskorski, Nicolas Stefanovitch, Brian Doherty, Jens P Linge, Sopho Kharazi, Jas Mantero, Guillaume Jacquet, Alessio Spadaro, and Giulia Teodori. 2023. Multi-label infectious disease news event corpus. In *Proceedings of the Text2Story'23 Workshop*.

Hristo Tanev, Jakub Piskorski, and Martin Atkinson. 2008. Real-time news event extraction for global crisis monitoring. In *Natural Language and Information Systems: 13th International Conference on Applications of Natural Language to Information Systems, NLDB 2008 London, UK, June 24-27, 2008 Proceedings 13*, pages 207–218. Springer.

Hristo Tanev, Vanni Zavarella, Jens Linge, Mijail Kabadjov, Jakub Piskorski, Martin Atkinson, and Ralf Steinberger. 2009. Exploiting machine learning techniques to build an event extraction system for portuguese and spanish. *Linguamática*, 1(2):55–66.

Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. 2023. Trigger or not trigger: Dynamic thresholding for few shot event detection. In *European Conference on Information Retrieval*, pages 637–645. Springer.

Marco A Valenzuela-Escárcega, Gus Hahn-Powell, Mihai Surdeanu, and Thomas Hicks. 2015. A domain-independent rule-based framework for event extraction. In *Proceedings of ACL-IJCNLP 2015 system demonstrations*, pages 127–132.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 998–1008, Minneapolis, Minnesota. Association for Computational Linguistics.

Zizhen Wang, Yixing Fan, Jiafeng Guo, Liu Yang, Ruqing Zhang, Yanyan Lan, Xueqi Cheng, Hui Jiang, and Xiaozhao Wang. 2020. Match^2: A matching over matching model for similar question identification. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 559–568.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

A Approximate pattern matching and eventness calculation algorithm

Given a set of event detection patterns
$Patterns = p_1, p_2, \ldots, p_n$
and a sentence s, the approximate pattern recognition calculates how likely it is that s contains an event, which we call *sentence eventness*. The approximate matching and the related *eventness* calculation happen in the following steps:

1. Encode each pattern p_i via a sequence of contextualized embedding vectors, using BERT: For each token from the pattern we take the contextualized word embedding vector from the *last layer of the encoder network*. Thus, we have a *sequence of context embedding vectors* with the length of the number of tokens in p_i. If, for example, the pattern is "protesters demand", the sequence will consist of two embedding vectors, one for each word. Lets call this *context embedding sequence* $es(p_i)$. Similarly, we obtain a sequence of embedding vectors for the sentence s, $es(s)$.

2. From the target sentence s, generate all the 2 to 20-grams which start and finish with a non-stop word, lets denote these n-grams with $ngrams(s)$. As an example, consider the sentence s ="The crowd in Damascus shouted slogans.", the $ngrams(s)$ will consists of the following ngrams: "crowd in Damascus", "crowd in Damascus shouted", crowd in Damascus shouted slogans", "Damascus shouted", "Damascus shouted slogans", and "shouted slogans".

3. For each n-gram $ng \in ngrams(s)$ we obtain the contextualized BERT embeddings $ses(ng, s)$: Note, we do not pass the ng to BERT for calculating the contextualized embedding vectors, but rather we take the corresponding embedding vectors from the embedding vector sequence of the whole sentence $es(s)$, ensuring better contextualization.

   In the example above, the vector sequence obtained from "crowd in Damascus" will be obtained as a subsequence (namely, the second, third and fourth vector) of the embedding vector sequence $es(s)$ of the full sentence "The crowd in Damascus shouted slogans".

4. Finally, we find the similarity of the embedding vector sequence of each pattern $es(p)$ with the embedding vector sequence of each n-gram, $ses(ng, s), ng \in ngrams(s)$. Then, we take as sentence eventness the maximal similarity between a pattern and an n-gram embedding sequence.

   We denote as $esim(es(p), ses(ng, s)$ the similarity of the embedding sequences $es(p)$ and $ses(ng, s)$. The eventness of the sentence, $ev(s)$, is calculated with the following formula:

   $ev(s) = \max_{p,ng} esim(p, ng)$, where $p \in Patterns, ng \in Ngrams(s)$.

5. If $ev(s) > \alpha$, then s is considered a sentence containing an event. The threshold α is being set empirically.

A.1 Calculating $esim$, similarity of a pattern and an n-gram embedding vector sequences

1. In order to compare how similar a pattern like p ="protesters demand" is similar to a n-gram like ng ="crowd in Damascus shouted slogans", our model first builds two sequences of embeddings corresponding to the two phrases: $es(p)$ and $ses(ng, s)$.

2. Then, the algorithm finds for each word in the pattern p the most similar word from ng, using the cosine similarity between the corresponding embedding vectors. In our example, the most similar word from ng for the word "protesters" is "crowd" and the most similar to "demand" is "slogans".

   We call these pairs of matching words *matching-pairs*: In the example above, they form the following set: $\{(protesters, crowd), (demand, slogans)\}$.

3. Then, we calculate the similarity between each pair of matching words and find their normalized sum, as it is shown in the formula on Figure 3: It is based on the sum of the similarities of the matching words, the inverse document frequency of the pattern words, and the difference of the positions of the matching words. In case of perfect similarity, equal pattern and event phrase, similarity function returns the value of 1.

$$esim(es(p), ses(ng, s)) = \frac{\sum_{(wp,wn)\in matching-pairs} cos(CE(wp), CE(wn)).idf(wp).\sqrt{\frac{1}{1+\delta(wp,wn)}}}{\sum_{(wp,-)\in matching-pairs} idf(wp)}$$

$matching - pairs$ - the set of pairs of words - first from the pattern p, the second from the n-gram ng, such that each word from p is paired with its most similar from ng, considering the cosine between their embedding vectors
$CE(w)$ - contextualised embedding vector of a word w
$idf(w)$ - inverse document frequency
$\delta(wp, wn)$ - the difference in the positions of the matching words

Figure 3: Similarity between pattern p and an ngram ng