mathematics

MDPI

Article

Identifying the Structure of CSCL Conversations Using String Kernels

Mihai Masala [1,2,*], Stefan Ruseti [1], Traian Rebedea [1], Mihai Dascalu [1,3], Gabriel Gutu-Robu [1] and Stefan Trausan-Matu [1,3]

[1] Computer Science and Engineering Department, University Politehnica of Bucharest, 313 Splaiul Independentei, 060042 Bucharest, Romania; stefan.ruseti@upb.ro (S.R.); traian.rebedea@upb.ro (T.R.); mihai.dascalu@upb.ro (M.D.); gabriel.gutu@upb.ro (G.G.-R.); stefan.trausan@upb.ro (S.T.-M.)
[2] 'Simion Stoilow' Institute of Mathematics of the Romanian Academy, 21 Calea Grivitei, 010702 Bucharest, Romania
[3] Academy of Romanian Scientists, Str. Ilfov, Nr. 3, 050044 Bucharest, Romania
* Correspondence: mihai_dan.masala@upb.ro

Abstract: Computer-Supported Collaborative Learning tools are exhibiting an increased popularity in education, as they allow multiple participants to easily communicate, share knowledge, solve problems collaboratively, or seek advice. Nevertheless, multi-participant conversation logs are often hard to follow by teachers due to the mixture of multiple and many times concurrent discussion threads, with different interaction patterns between participants. Automated guidance can be provided with the help of Natural Language Processing techniques that target the identification of topic mixtures and of semantic links between utterances in order to adequately observe the debate and continuation of ideas. This paper introduces a method for discovering such semantic links embedded within chat conversations using string kernels, word embeddings, and neural networks. Our approach was validated on two datasets and obtained state-of-the-art results on both. Trained on a relatively small set of conversations, our models relying on string kernels are very effective for detecting such semantic links with a matching accuracy larger than 50% and represent a better alternative to complex deep neural networks, frequently employed in various Natural Language Processing tasks where large datasets are available.

Keywords: Natural Language Processing; educational technology; neural networks; CSCL conversations; string kernels

Citation: Masala, M.; Ruseti, S.; Rebedea, T.; Dascalu, M.; Gutu-Robu, G.; Trausan-Matu, S. Identifying the Structure of CSCL Conversations Using String Kernels. *Mathematics* **2021**, *9*, 3330. https://doi.org/10.3390/math9243330

Academic Editors: Florentina Hristea and Cornelia Caragea

Received: 10 November 2021
Accepted: 17 December 2021
Published: 20 December 2021

1. Introduction

With an increased prevalence of online presence, accelerated by the current COVID-19 pandemic [1], online messaging applications are gaining an increased popularity. Online social networks make a significant percentage of these platforms, but standalone chat applications are also widely adopted. These platforms are not used only for entertainment purposes, but their applications cover various activities, including and even promoting collaborative learning and creative thinking [2].

Artificial Intelligence techniques have been widely employed in various educational settings [3,4], ranging from classifying learning styles [5,6], to finding active collaborators within a group [7], to providing personalized feedback [8,9], and even customizing curriculum content [10]. Student learning styles (Diverger, Assimilator, Converger or Accommodator using Kolb's Learning Style Inventory [11]) can be precisely classified based on learner's EEG waves and further correlated with IQ and stress levels [5,6]. In collaborative learning settings, automated systems classify students based on their implication and collaboration activity, and provide information to support and enhance students' involvement in key moments [7]. Moreover, online interaction patterns changed in the current pandemic context, both in online learning environments [12] and in general, with an

increased online participation, whose traces can be effectively used to create even more advanced predictive models.

Education leans on online communication to enhance the learning process by integrating facilities, such as discussion forums or chat conversations, to stimulate collaboration among peers and with course tutors [13]. Nevertheless, group size has a major impact on delivery time, level of satisfaction, and overall quality of the result [14]. Usually, these conversations occur between more than two participants, which leads to numerous context changes and development of multiple discussion threads within the same conversation [15,16]. As the number of participants increases, the conversation may become harder to follow, as the mix of different discussion threads becomes more frequent. Moreover, divergences and convergences appear between these threads, which may be compared to dissonances and consonances among counterpointed voices in polyphonic music, which have a major role in knowledge construction [15,16].

Focusing on multi-participant chat conversations in particular, ambiguities derived from the inner structure of a conversation are frequent due to the mixture of topics and of messages on multiple discussion threads, that may overlap in short time spans. As such, establishing links between utterances greatly facilitates the understanding of the conversation and improves its readability, while also ensuring coherence per discussion thread. Applications that allow users to manually annotate the utterances they are referring to, when writing their reply, have long existed [17], whereas popular conversation applications (e.g., WhatsApp) successfully integrated such functionalities. Although users are allowed to explicitly add references to previous utterances when issuing a reply, they do not always annotate their utterances, as this process feels tedious and interrupts the flow of the conversation.

Our research objective is to automatically discover semantic links between utterances from multi-participant chat conversations using a supervised approach that integrates neural networks and string kernels [18]. In terms of theoretical grounding, we establish an analogy to the sentence selection task for automated question answering—in a nutshell, detecting semantic links in chat conversations is similar, but more complex. In question answering, most approaches [19–22] consider that the candidate sentence most similar to the question is selected as the suitable answer. In our approach, the user reply is semantically compared to the previous utterances in the conversation, and the most similar contribution is selected while considering a sliding window of previous utterances (i.e., a predefined time-frame or using a preset number of prior utterances). It is worth noting that we simplified the problem of identifying links between two utterances by reducing the context of the conversation to a window of adjacent utterances. Nevertheless, we emphasize the huge discrepancy in terms of dataset sizes between the question answering task and the small collections of conversations currently available for our task.

We summarize our core contributions as follows:

- Employing a method grounded in string kernels used in conjunction with state of the art NLP features to detect semantic links in conversations; in contrast to previous studies [23,24], we also impose a threshold for practical usage scenarios, thus ensuring the ease of integration of our model within chat environments;
- Providing extensive quantitative and qualitative results to validate that the lexical information provided by string kernels is highly relevant for detecting semantic links across multiple datasets and multiple learning frameworks (i.e., classification and regression tasks);
- Obtaining state of the art results on two different datasets by relying on string kernels and handcrafted conversation-specific features. Our method surpasses the results of Gutu et al. [25,26] obtained using statistical semantic similarity models and semantic distances extracted from the WordNet [27] ontology. In addition, our experimental results argue that simpler supervised models, fine-tuned on relatively small datasets, such as those used in Computer-Supported Collaborative Learning (CSCL) research,

may perform better on specific tasks than more complex deep learning approaches frequently employed on large datasets.

In the following subsections we present state-of-the-art methods for computing text similarity and for detecting semantic links.

1.1. Lexical and Semantic Models for Text Similarity

Early models to compute semantic similarity consider semantic distances (e.g., path length, Wu-Palmer [28], or Leackock-Chodorow [29]) in lexicalized ontologies, namely WordNet [27], as well as Latent Semantic Analysis (LSA) [30]. LSA uses a term-document matrix which stores the number of occurrences of each term in every document. Singular Value Decomposition followed by a projection on the most representative dimensions is then performed to transform the matrix into a latent semantic space. Semantic similarity scores between words are calculated using cosine similarity scores within this semantic vector space.

1.1.1. Word Embeddings

Word embeddings represent words in a vector space using their context of occurrence within a corpus. Among existing models, word2vec is one of the most frequently used embeddings methods. Word2vec uses distributed word embeddings computed using a simple neural network that considers the context of words as n-gram co-occurrences [31]; the similarity between two texts is determined using cosine similarity. Word2vec, in the skip-gram framework, is in fact a generative neural model [32] trained to predict the words that appear in the context of a given word. Another popular model, Glove [33], computes word embeddings using an approach based on a count-based approach, using the number of occurrences of any two words within a text. Both models rely on word level representations for texts. Another approach is offered by FastText [34] which uses character n-grams as an extension of word2vec and thus is able to compute character n-grams embeddings. This method is very useful for determining embeddings for out-of-vocabulary words, e.g., words that do not appear or are not very frequent in the corpora used for training the embeddings space.

1.1.2. String Kernels

String kernels [35] are functions used at the character level. The underlying assumption is that a satisfactory similarity measure between two documents can be associated with the number of shared sub-strings of a predefined size. Instead of representing texts in this sub-string induced space, string kernels use a function which replicates the dot-product of two texts in this high-dimensional space. The higher the value of the kernel function, the more similar the texts are.

Variations in the size of n-grams (commonly between 2 and 10 characters) enable the generation of different string kernels. String kernels also vary depending on the function used for computing the overlap between two texts. The most common string kernels are spectrum, intersection, and presence [36]. Spectrum is calculated as the dot-product between the frequencies of n-grams (Equation (1)). Intersection kernel relies on the minimum of the two frequencies (Equation (2)). The presence kernel encodes whether an n-gram is present or not in a string by using presence bits (Equation (3)). In our experiments, normalized versions of these kernels were used.

$$k_p^s(a,b) = \sum_{v \in \Sigma^p} num_v(a) \cdot num_v(b) \tag{1}$$

$$k_p^{\cap}(a,b) = \sum_{v \in \Sigma^p} min\{num_v(a), num_v(b)\} \tag{2}$$

$$k_p^{0/1}(a,b) = \sum_{v \in \Sigma^p} in_v(a) \cdot in_v(b) \tag{3}$$

where:

- Σ^p = all p-grams of a given size p,
- $num_v(s)$ = number of occurrences of string (n-gram) v in document s,
- $in_v(s)$ = 1 if string (n-gram) v occurs in document s, 0 otherwise.

String kernels can also be used as features for different classifiers to solve tasks such as native language identification [37], protein fold prediction, or digit recognition [38]. Beck and Cohn [39] use the Gaussian Process framework on string kernels with the goal of optimizing the weights related to each n-gram size, as well as decay parameters responsible for gaps and matches. Their results show that such a model outperforms linear baselines on the task of sentiment analysis. Another important result is that, while string kernels are better than other linear baselines, non-linear methods outperform string kernels; thus, non-linearly combining string kernels may further improve their performance. One such extension was proposed by Masala et al. [18] for the task of question answering. The authors show that a shallow neural network based on string kernels and word embeddings yielded good results, comparable to the ones obtained by much more complex neural networks. The main advantage of the approach is that a small number of parameters needs to be learned, which allows the model to be also trained and used on small datasets, while concurrently ensuring a very fast training process. We rely on a similar approach for detecting semantic links in chat conversations, a task with significantly smaller datasets than question answering.

1.1.3. Neural Models for Text Similarity

Neural-based models have been widely used for computing similarity between paragraphs for question answering tasks [21,40,41]. Given a question and a list of candidate answers, the task of selecting the most reasonable answer is also known as answer selection (a sub-task of question answering). The general approach for computing the similarity between two text sequences is the following: compute an inner representation for both text segments using a neural network and then apply a similarity function (which, in turn, can be modelled with a neural network). Common neural models used for answer selection include Bidirectional Long Short-Term Memory (Bi-LSTM) [42] or Convolutional Neural Networks (CNN) [43]. Because there is no restriction on the length of the analyzed sentences, the dynamic number of outputs of the Bi-LSTM must be converted into a fixed-length representation. This transformation can be performed by simple average or max pooling, concatenation of the first and the last output, or by more complex methods such as applying another CNN layer on top of these inner representations [41].

In addition, attention mechanisms are frequently employed in neural network models as they enable long-range dependencies between parts of the input [44,45]. In the context of question answering, the attention mechanism allows the model to also take into account the question, when computing the representation of a candidate answer. Intuitively, attention allows the model to peek at the question when computing the representation of the answer, thus providing the ability to better focus on the relevant parts of the answer (with regards to the question). Dos Santos et al. [21] proposed the usage of an attention mechanism that allows both the question and the answer to influence each others' representation. After computing the inner representations of the question and answer using either a Bi-LSTM or a CNN, the authors combined the representations into a single fixed-size matrix from which attention weights are extracted, and afterwards used to compute the final representations of the question and answer.

Bachrach et al. [40] proposed the usage of a global view of the question together with its inner representation, when computing the attention weights. One way of obtaining such global information from the question is to use a multi-layer perceptron (MLP) on the bag-of-words representation of the question. Wang et al. [46] explored the use of comparison functions (e.g., element-wise subtraction and multiplication, a simple MLP) for combining the attention-weighted representation of the question with the representation of the answer, followed by a CNN for the final classification step.

Transformer-based architectures [47] have become popular in the NLP domain because of their state-of-the-art performance on a wide range of tasks [48–53]. The idea behind the Transformer architecture was to replace the classical models used for processing sequences (e.g., RNNs or CNNs) with self-attention mechanisms that allow global and complex interactions between any two words in the input sequence. For example, BERT [48] used multiple layers of Transformers trained in a semi-supervised manner. The training of BERT is based on two tasks: Masked LM (MLM)—in which a random token (word) is masked and the model is asked to predict the correct word—and Next Sentence Prediction (NSP)—in which the model is given two sentences A and B and is trained to predict whether sentence B follows sentence A. In our experiments, we consider the NSP pretrained classifier.

1.2. Detection of Semantic Links

The manual annotation of semantic links is a time consuming and difficult task. Although many chat applications provide the possibility to explicitly introduce such links, participants frequently forget or do not think it is necessary to add links to the referred contribution, as the process breaks the conversation flow. Techniques for automated annotation of such links were previously developed and were referred to as *implicit links detection* [54] or *chat disentanglement* [55–58].

1.2.1. Semantic Distances and Semantic Models

Previous experiments by Gutu et al. [25,26] considered semantic distances between utterances in a floating window and statistical semantic similarity models trained on large corpora of documents. The authors explored the optimal window sizes in terms of the distance (expressed as count of intermediary utterances) and time spent between two utterances in order to search for semantic links. For a given reply, the contribution with the highest semantic score from the window was chosen as the referred utterance. The performance of this approach in detecting semantic links was evaluated based on the explicitly referred links added by participants from a conversation [25]. A corpus of 55 CSCL chats with multiple participants was used for this experiment. The same corpus was used in the current study and is detailed later on in Section 2.1.1.

1.2.2. Neural Networks

Long Short-Term Memory (LSTM) [59] networks have been used to capture the message-level and context-level semantics of chat utterances with the end goal of disentanglement [57]. Jiang et al. [56] proposed a two stage method with a NN backbone for chat disentanglement. In the first stage, a Siamese Hierarchical Convolutional Neural Network (SHCNN) is used for estimating the similarity of two utterances that was further used to establish the disentangled conversations. Li et al. [58] used Transformer-based architectures [47] to detect semantic links. Their model considered Bidirectional LSTM networks on top of a BERT model to capture the intricate interactions between multiple utterances and, finally, to identify pairs of related utterances. We emphasize that, especially in the case of state-of-the art NLP architectures, a significant amount of data is required to properly train the aforementioned models.

Besides the textual content of a contribution, previous experiments by Masala et al. [23,24] used additional meta-information to discover semantic links using NN classifiers. Such conversation-specific meta information included the time spent between two utterances, the distance between them, or whether the two utterances belonged to the same author [23,24]. Mehri et al. [60] also employed Recurrent Neural Networks (RNNs) for modelling semantic relationships between chat utterances. Semantic information was used together with meta-information (such as distance between utterances) for thread partitioning and the detection of direct replies in conversations.

1.2.3. Other Computational Approaches

Previous work by Trausan-Matu and Rebedea [54] considered speech acts [61] for identifying continuations or question answering patterns between utterances. Moldovan et al. [62] argued that speech acts can be determined with a high accuracy by only using the first few words in the contribution. Moreover, the dialogue between participants can highlight patterns that may be automatically identified. In an educational context [63], student profiles were created by analyzing the interactions between the teacher and the student, as well as the posts in the discussion forums.

2. Method

2.1. Datasets

2.1.1. Corpus of CSCL Chat Conversations

Our experiments were performed on a collection of 55 chat conversations (ChatLinks dataset, available online at https://huggingface.co/datasets/readerbench/ChatLinks, accessed on 10 November 2021) held by Computer Science undergraduate students [15,25]. Participants had to discuss on software technologies that support collaborative work in a business environment (e.g., blog, forum, chat, wiki). Each student had to uphold one preferred technology in the first part of the conversation, introducing benefits or disadvantages for each CSCL technology, followed by a joint effort to define a custom solution most suitable for their virtual company in the second part of the chat. The discussions were conducted using ConcertChat [17], a software application which allows participants to annotate the utterance they refer to, when writing a reply. The vision behind these interactions was grounded in Stahl's vision of group cognition [64] in which difficult problems can be solved easier by multiple participants using a collaborative learning environment.

Two evaluations were considered. The first one relies on the *exact matching* between two utterances, which checks whether the links are identical with the references manually added by participants while discussing. The second approach considers *in-turn matching* which checks whether the detected links belong to the same block of continuous utterances written by the same participant, as defined in the manually annotated references. The automated approach computes similarity scores between each given contribution and multiple previous utterances, within a pre-imposed window size. The highest matching score is used to establish the semantic link. A conversation excerpt depicting an exact matching between the reference and semantic link is shown in Table 1. An in-turn matching example is shown in Table 2. In both cases, the emphasized text shows the utterance which denotes the semantic link. The Ref ID column shows the explicit manual annotations added by the participants.

Table 1. Fragments extracted from conversations showing exact matching (Semantic link is highlighted in bold).

Utt. ID	Ref. ID	Speaker	Content
257		**Razvan**	**High-activity forum threads can be automatically taken to chat if sufficient users are online**
258		Bogdan	I think it's a good ideea let the user post their pictures, their favorite books, movies
259	257	Andreea	and if the time between posts is very short
260	**258**	**Andreea**	**a user profile, of course**
261		Bogdan	that way big communities will be created
262	**258**	**Razvan**	**personal, social blogs, chatrooms and forums besides educational ones**

Table 1. *Cont.*

Utt. ID	Ref. ID	Speaker	Content
177		**Oana**	**i belive that on forums, you can also show that "human" part :)**
178	177	Tibi	yes ...but you cannot build a relationship
179		Tibi	a long term relationship
180		**Oana**	**With who?**
181		Oana	With other people?
182		Oana	Why's that?
183		Oana	You can interact with anybody.
184		Oana	You post a message, about a topic
185		Oana	Furthermore, other people can answer to that message or say an opinion...
186	180	Tibi	as i said before...people became attached of your writing...they want to descover more of you...
187		Oana	It's a kind of conversation
188		**Oana**	**Furthermore, any conversation can lead to a relation**
189	**188**	**Tibi**	**yes...but on a certain topic only...**
190		Oana	Yes, well, I have my own kind of writting :)
191	189	Oana	yes, but if you want, you can go and change that topic

Table 2. Fragments extracted from conversations showing in-turn matching (Semantic link is highlighted in bold).

Utt. ID	Ref. ID	Speaker	Content
107		Lucian	They do not require any complicate client application, central mediation
108		**Lucian**	**Actually, all this arguments are pure technical**
109		Lucian	The single and best reason for which chats are the best way of communication in this age of technology is that
110		Lucian	Chat emulate the natural way in which people interact. By talking, be argumenting ideas, by shares by natural speech
111		Lucian	Hence,chat is the best way to transform this habit in a digital era.
112		Lucian	We can start debating now? :D
113	111	Florin	I would like to contradict you on some aspects
379		Alina	No, curs.cs is an implementation of moodle
380		Alina	Moodle is jus a platform
381		Alina	You install it on a servere
382		**Alina**	**and use it**
383		Alina	Furthermore, populate it wih information.
384		Andreea	and students are envolved too in development of moodle?
385		Alina	It has the possibility of wikis, forums, blogs. I'm not sure with the chat, though.
386	379	Stefan	Yes that is right

The manually added links were subsequently used for determining accuracy scores for different similarity metrics, using the previous strategies (e.g., exact and in-turn matching). The 55 conversations from the corpus made up to 17,600 utterances, while 4500 reference links were added by participants (e.g., about 29% of utterances had a corresponding reference link). Out of the 55 total conversations, 11 of them were set aside and used as a test set.

A previous study by Gutu et al. [25] showed that 82% of explicit links in the dataset were covered by a distance window of 5 utterances; 95% of annotations were covered by enlarging the window to 10 utterances, while a window of 20 utterances covered more than 98% of annotated links, while considering time-frames, a 1 min window contained only 61% of annotations, compared to the 2 min window which contains about 77% of all annotated links. A wider time-frame of 3 min included about 93% of all links, while a

5 min window covered more than 97% of them. As our aim was to keep the majority of links and to remove outliers, a 95% coverage was considered ideal. Smaller coverages were included in our comparative experiments. Thus, distances of 5 and 10 utterances were considered, while time-frames of 1, 2, and 3 min were used in the current experiments.

2.1.2. Linux IRC Reply Dataset

Besides the previous collection of chat conversations, we also used the chat conversations dataset proposed by Mehri et al. [60] for classifying pairs of utterances. This dataset was specifically built to capture direct reply relationships between utterances. The data consists of a subset of '#Linux IRC log data' [55], manually annotated with direct reply relationships. Volunteers, familiar with Linux, were instructed to go through the chat messages and select the immediate parents for every message. A message might have no parents (e.g., when starting a new conversation thread) or it might have multiple parents (e.g., an answer to a multi-participant thread). On average, a message had 1.22 direct parents and 1.70 direct children in this dataset, while this dataset is not related to formal education, it is a great example of using chats in communities of practice in the real world (e.g., specialists working on Linux).

2.2. Neural Model for Semantic Links Detection

One of our key insights is that answers connect to questions in a similar manner to how semantic links connect utterances, in the sense of information flow or continuation of ideas. Therefore, we theorize that answer selection methods can be effective in detecting semantic links. We adapt the model introduced by Masala et al. [18] for answer selection to our task. Figure 1 presents the processing flow. The goal of our model is to combine lexical features (in the form of string kernels) with semantic and conversation-specific information to better capture semantic links between utterances.

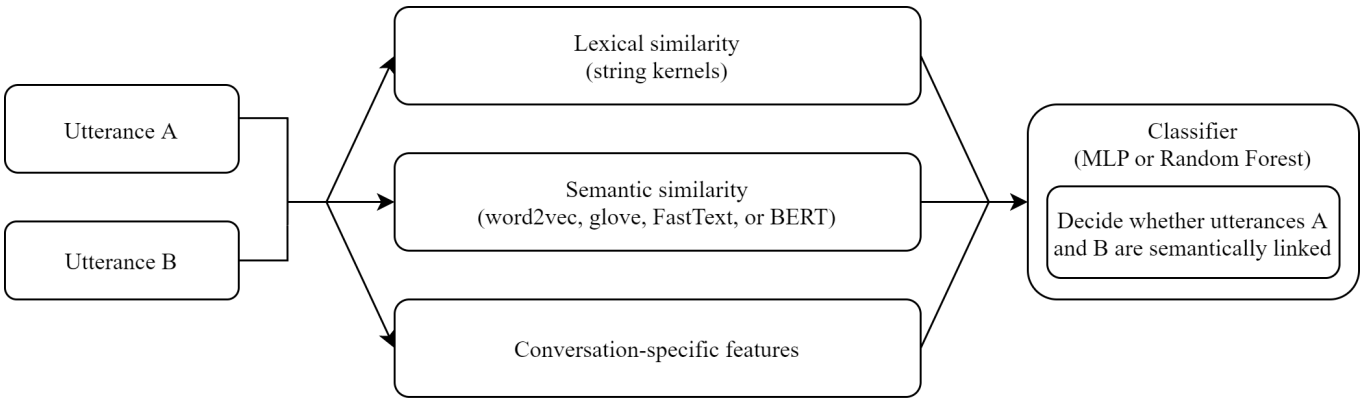

Figure 1. Conceptual diagram of our approach.

Moreover, we establish strong supervised and unsupervised baselines for evaluating our approach, namely:

- Path Length [25]: Previous best results for detecting semantic links were achieved on the same dataset in an unsupervised manner by using WordNet Path Length as similarity distance. Path Length is based on the shortest length path between two concepts in the WordNet ontology.
- String Kernels: We use string kernels as a measure of similarity; we experiment with intersection, presence and spectrum kernels [36], on a 3–7 g range .
- AP-BiLSTM [21]: A supervised method which achieves top results on the answer selection task. Both utterances are passed through a Bidirectional LSTM network. Attention vectors are extracted from the hidden states (at each time step), leading to

attention-based representations for both utterances. Cosine similarity is then used on the attention-weighted utterances for computing the similarity between them.

- BERT [48]: We use a pretrained BERT-base model to compute the probability that two utterances follow one another, as a continuation of ideas. For this task, we fine-tune BERT, following the approach proposed by Devlin et al. [48]. Therefore we optimize the binary cross-entropy loss with Adam optimizer [65] with a learning rate of 1×10^{-5}, using batches of size 32 for 7 epochs. All hyperparameters were selected using 10-fold cross-validation.

Three different types of string kernels (spectrum, presence and intersection) were considered in the proposed supervised neural model, each with five character n-gram ranges: 1–2, 3–4, 5–6, 7–8, and 9–10. Hence, we compute 15 similarity scores for each pair of utterances, and we combine the above-mentioned features using a simple feed-forward multilayer perceptron (MLP) with one hidden layer. The MLP computes a single number for each pair of utterances, namely a similarity score. The hidden layer size was set to 8 for all follow-up experiments, while the batch size was fixed at 100. Hinge loss (Equation (4)) was used as objective function, similar to the loss proposed by Hu and Lu [66] for finding similarities between two sentences. The margin M was set to 0.1 and minimized using the Adam optimizer [65]. The previous utterance most similar to the current one is selected as the semantic link by our model, as well as for the baselines.

$$e(u_r, u^+, u^-) = max(0, M + sim(u_r, u^-) - sim(u_r, u^+)) \quad (4)$$

where:

- u_r refers to the utterance for which the link is computed,
- u^+ refers to the manually annotated utterance,
- u^- is an incorrect utterance contained within the current window,
- $sim(u_r, u)$ refers to the semantic similarity score calculated by the MLP between the two utterances representations,
- M is the desired margin among positive and negative samples.

Furthermore, we enhance the lexical information with conversation-specific information, including details regarding the chat structure, as well as semantic information. The conversation-specific features are computed for each candidate contribution (for a link) as follows: we check whether the contribution contains a question, or the candidate and the link share the same author, while referring to the chat structure, we use the number of in-between utterances and the time between any two given utterances. Two methods for computing semantic information were considered. Given two utterances, the first method computes the embedding of each utterance as the average over the embeddings (e.g., pretrained word2vec, FastText and GloVe models) of all words from the given utterance, followed by cosine similarity. The second method relies on BERT, namely the Next Sentence Prediction classifier, which is used to compute the probability that the two utterances follow one another.

2.3. Detecting Direct Replies in Linux IRC Chats

To further validate the effectiveness of string kernels in modeling chat conversations, we investigate the use of string kernels as a feature extraction method on a different dataset and on a slightly different task. Instead of framing the explicit link detection problem as regression (e.g., given an utterance, find its semantic link by computing a score), we treat it as a classification (e.g., a binary classification to establish whether two utterances are connected or not). Starting from the approach proposed by Mehri et al. [60], we train a classifier which outputs, for two given utterances, the probability of the first one being a reply to the second message. Features extracted from string kernels are also considered, resulting in three categories of features (see Table 3): conversation-specific features, semantic information, and lexical information. We further note that for this task, we aim to replicate as accurately as possible the Mehri's et al. [60] approach by using the

same model architecture, features, and training methods, the only difference being that we investigate the usage of string kernels and BERT-based features.

Table 3. Features used in the reply classifier.

Conversation	Time	Time difference between the two utterances (in seconds)
	Distance	The number of messages between the two utterances
	Same author	Whether two utterances have the same author
	Mention child	Whether the parent message mentions the author of the child message
	Mention parent	Whether the child message mentions the author of the parent message
Semantic	RNN output	Probability outputted by the RNN
	BERT	Probability outputted by BERT NSP
Lexical	String Kernels	Similarities given by string kernels

We consider two approaches for capturing semantic information. The first model considers an RNN trained on the Ubuntu Dialogue Corpus, as proposed by Lowe et al. [67]. The purpose of this network is to model semantic relationships between utterances and it is trained on a large dataset of chat conversations. We use a siamese Long Short-Term Memory (LSTM) [59] network to model the probability of a message following a given context. Both the context and the utterance are processed first using a word embedding layer with pre-trained GloVe [33] embeddings, which are further fine-tuned. After the embedding layer, the representations of context and utterance are processed by the LSTM network. Let c and r be the final hidden representation of the context and of the utterance, respectively. These representations, alongside a learned matrix M, are used to compute the probability of a reply (see Equation (5)).

$$P(reply|context) = \sigma(c^T Mr) \tag{5}$$

The same training procedure introduced by Lowe et al. [67] is applied, namely minimizing the cross-entropy of all labeled (context, contribution) pairs by considering a hidden layer size of 300, an Adam optimizer with gradients clipped to 10, and a 1:1 ratio between positive examples and negative examples (that are randomly sampled from the dataset). In our implementation, a dropout layer was also added after the embedding layer and after the LSTM layer for fine-tuning the embeddings, with the probability of dropping inputs set to 0.2.

The second method is more straightforward and it is based on BERT [48]. We use a pretrained BERT-base model and we query the model for whether two utterances are connected, using the Next Sentence Prediction classifier.

Furthermore, we employ string kernels as a feature extraction method for lexical information. Given a pair of utterances, we compute their lexical similarity with three string kernels (spectrum, presence and intersection) at different granularity (n-gram ranges): 1–2, 3–4, 5–6, 7–8 and 9–10. Thus, a lexical feature vector $v \in \mathcal{R}^{15}$ is computed for each pair of messages.

Mehri's et al. [60] training methodology was used, namely the Linux IRC reply dataset was split into a set of positive examples (annotated replies) and a set of negative examples (the complement of the annotated replies). This leads to a very imbalanced dataset, with most pairs being non-replies. One method to alleviate this problem is to only consider pairs of message within a time frame of 129 s [60] one from another. Different classifiers relying on the features described in Table 3 were trained using 10-fold cross-validation. De-

cision trees, a simple Multi-Layer Perceptron (MLP) with a hidden size of 20, and random forest [68] were considered in our experiments, each with their benefits and drawbacks.

3. Results

The following subsections provide the results on the two tasks, namely semantic links and direct reply detection, arguing their strong resemblance in terms of both models and features. Subsequently, we present a qualitative interpretation of the results.

3.1. Semantic Links Detection in the CSCL Chat Conversations

We first evaluate our approach on the CSCL chats dataset described in Section 2.1.1. Our supervised neural network is compared with state-of-the-art methods for answer selection and semantic links detection. We also compare our model with an unsupervised method based only on string kernels. For this baseline, the considered n-gram range (3–7) was selected to maximize the accuracy on a small evaluation set. All supervised methods are trained and evaluated using 10-fold cross-validation. Note that results are reported on the test set.

The following pretrained embeddings were used in our experiments: word2vec, FastText, and GloVe. The word2vec [32] embeddings were pretrained on the Google News Dataset. The FastText embeddings [34] were pretrained on Wikipedia, whereas the GloVe embeddings [33] were pretrained on a Wikipedia 2014 dump and Gigaword 5 (https://catalog.ldc.upenn.edu/LDC2011T07 accessed on 10 November 2021). All these pretrained models are publicly available and widely used in NLP research.

Results are presented in Table 4. The first part includes the accuracy obtained by the baseline methods, while the AP-BiLSTM is a top performing model in the task of answer selection, it performs about the same as the unsupervised path length method for our specialized task of detecting semantic links. AP-BiLSTM obtains even worse performance on the in-turn metric. This is due to the small size of the training dataset compared to the large number of parameters of the model. The BERT-based method outperforms other baselines by a significant margin on every window-time frame, but its performance degrades with larger windows. One possible reason is that the BERT model is pretrained on English Wikipedia and BookCorpus which contain longer sentences, especially when compared to the utterances from the first dataset, while we fine-tune the entire model, its small number of samples cannot alleviate this problem.

Table 4. Results for semantic links detection (Exact matching accuracy (%)/In-turn matching accuracy (%).

Window (Utterances)	**5**			**10**		
Time (mins)	**1**	**2**	**3**	**1**	**2**	**3**
Path Length [25]	32.44/41.49	32.44/41.49	not reported	31.88/40.78	31.88/40.78	not reported
AP-BiLSTM [21]	32.95/34.53	32.39/35.89	33.97/37.58	33.86/35.10	28.89/31.82	24.49/28.32
Intersection kernel	31.40/34.59	33.87/39.58	33.58/40.01	31.71/34.78	32.24/37.66	29.47/35.24
Presence kernel	31.84/34.94	33.97/39.81	33.58/40.01	31.80/34.89	32.33/37.71	29.67/35.41
Spectrum kernel	31.21/34.34	33.45/39.12	33.17/39.49	31.39/34.46	31.56/36.72	28.75/34.26
BERT	40.07/41.99	43.45/47.07	44.47/48.42	40.41/42.33	41.53/45.15	38.15/38.60
NN using sk	35.21/36.90	35.55/39.39	35.77/39.95	35.55/37.02	34.08/37.47	30.24/33.74
NN using sk + conv	37.92/39.39	45.48/49.66	47.06/51.80	38.14/39.50	46.27/50.79	**47.85**/**52.93**
NN using sk+sem	36.45/38.14	36.90/40.47	36.00/40.29	36.68/38.14	35.10/38.26	31.26/34.76
NN using sk + sem + conv	37.02/38.60	46.38/50.00	48.08/52.25	37.24/38.71	47.29/51.46	**49.09**/**53.83**
NN using sk + BERT	37.35/38.71	39.16/42.21	39.61/43.00	37.24/38.60	37.13/40.18	33.52/36.90
NN using sk + BERT + conv	40.40/42.21	46.72/49.88	48.08/51.91	40.63/42.43	46.95/50.33	48.08/52.37

Note: sk—string kernels; conv—conversation-specific features, namely window and time (window—# of in-between utterances; time—elapsed time between utterances); question and author (question—whether the utterance contains a question; author—if the utterance shares the same author as the utterance containing the link); sem—semantic information. **Bolded** values represent the best results with and without semantic information.

Unsupervised string kernels by themselves provide inconclusive results as they are not always better than previous methods, but they do seem to work better especially for larger windows. Moreover, we find that there is no convincing difference between any of three types of string kernel functions.

The results obtained using the neural model are presented in the middle part of Table 4. We performed multiple experiments by introducing conversation-specific and semantic features independently (second and third row), and together (fourth row). Word2vec, FastText, and Glove (embedding size 100 and 300) are used for extracting semantic information, with no significant difference in results. Conversation-specific features provide a significant accuracy increase for the task of detecting semantic links. Previous studies found a similar conclusion, as the path length method uses the distance between two utterances as a weighting factor [25], while semantic information improves performance, we cannot consider that the gain is impressive.

In the last part of Table 4, we present the results obtained when extracting semantic information with the method based on BERT. However, despite BERT's Transformer model complexity, the results are similar with the word embeddings semantic encoding. Additional discussions of the results in Table 4 are presented in Section 4.

3.2. Imposing a Threshold for Practical Usage Scenarios

In order to make our system ready for use in practice, we propose a threshold in the following manner: (a) run our model for each utterance in the conversation, (b) pick the utterance with the highest similarity score, and (c) if this score is higher (or equal) than the threshold, recommend adding the semantic link in the conversation. If the similarity score is lower than the threshold, we consider that the given utterances are not linked to any previous utterance. Setting the threshold is of utmost importance: a too high value will likely generate a high number of false negatives, while a too low value would yield more false positives. All of the following experiments and results refer only to the exact match metric.

Overall, the absolute value of the similarity score obtained on a pair of utterances is not very relevant per se due to the minimization of the hinge loss. The similarity scores become relevant in context, when compared with each other (in a given window). For this reason, we cannot simply look at the absolute values generated by the model, we need to look at the relation between scores obtained by utterances in the same window. Therefore, we compute, for each of the 10 folds, the mean and standard deviation values across all predictions. Next, for each fold, we search for the threshold value that maximizes the F1 score. For each threshold value, we compute how far this value (in terms of standard deviation) is from the mean prediction value. This distance is computed independently for each fold (as the model is trained for each fold). The mean value of those distances is used for computing the threshold, then a model is trained on the entire training set and the following formula is used for establishing the final threshold (Equation (6)):

$$threshold = mean_p + mean_d * std_p \tag{6}$$

where:

- $mean_p$ is the mean of the predictions on the training set,
- std_p is the standard deviation of the predictions on the training set,
- $mean_d$ is the mean of the distances between the mean prediction and the best threshold found for each fold.

The final values are the following: $mean_p = 0.478$, $std_p = 0.14$, $mean_d = 1.32$, which lead to a $threshold = 0.66$. We evaluate our approach on the test set with the threshold set to 0.66 and obtain an F1 Score of 0.31 (for the positive class) and a 51.67% accuracy. In a practical scenario, if the semantic link suggestion is incorrect, the user might just ignore the predicted link.

3.3. Detecting Direct Replies

As previously mentioned in Section 2.3, we first train a siamese-LSTM on the large Ubuntu Dialogue Corpus [67] to predict whether an utterance follows a sequence of utterances. Our implementation slightly outperforms the results reported in [67], using as metric the 1 in 10 next utterance recall: 95.2% versus 92.6% for R@5, 80.0% versus 74.5% for R@5, and 65.4% versus 60.4% for R@1. For all following experiments, we consider our implementation of siamese-LSTM.

In Table 5, we present the results of the proposed methods on the Linux IRC reply dataset. To better understand how informative are string kernels, we made three sets of experiments for each classifier: (a) one set using conversation-specific and semantic features as proposed in [60] (with the addition of semantic information obtained by using pretrained BERT; see the first two columns in Table 5), (b) a second set of experiments including lexical features (presented in the middle half of Table 5), and (c) the last set where semantic information is replaced with the features extracted with string kernels (in the last column of Table 5. We used 250 trees for the random forest classifier and we adjusted the class weights to penalize false positive to better handle the imbalanced dataset. Specifically, the class weight of the positive class was 1, whereas the weight of the negative class was set to *#non_replies*/*#replies*.

We can observe in Table 5 that lexical information provided by string kernels can be combined with more complex features (such as semantic features obtained by siamese-LSTM) to improve performance. Out of all the classifiers, the random forest model performed the best (0.65 F1 score; 0.75 precision and 0.57 recall for the reply class).

Table 5. F1 scores (positive class) for reply detection on Linux IRC reply dataset. **C** denotes Conversation features, **S** denotes semantic information while **L** stands for lexical information. More details about used features can be found in Table 3.

Method	C + S(RNN)	C + S(BERT)	C + S(RNN) + L	C + S(BERT) + L	C + L
Decision Tree	0.53	0.54	0.54	0.56	0.55
MLP	0.60	0.60	0.62	0.61	0.60
Random Forest	0.59	0.62	0.63	**0.65**	0.63

In addition, we identify which features were the most important for the random forest model by considering Gini feature importance [68]. For the case in which we do not use lexical information, the semantic information is the most important feature (Gini value of 0.40), followed by time difference (Gini value of 0.27), and space difference (Gini value of 0.14); see Figure 2.

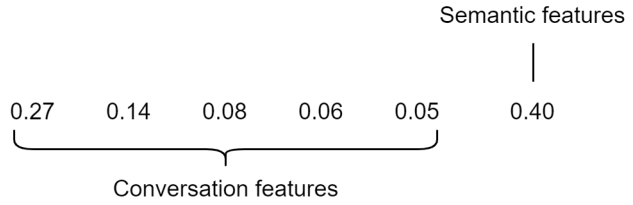

Figure 2. Gini feature importance for random forest without string kernels.

When adding string kernels, the semantic information is still the most important feature (Gini value of 0.16), followed by time and distance conversation features (Gini value of 0.15 and 0.12, respectively), and by string kernels on ngrams of size 1–2 (Gini value of 0.11); see Figure 3.

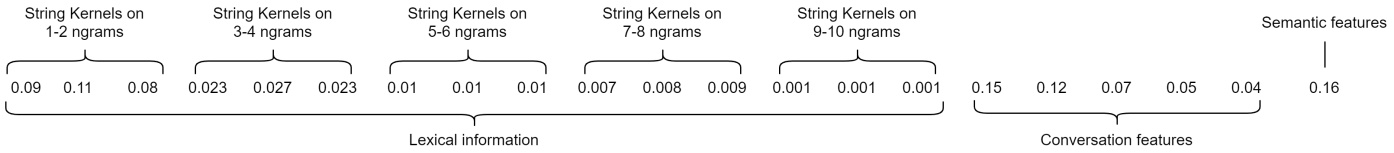

Figure 3. Gini feature importance for random forest with string kernels.

If the semantic information is replaced with lexical information, the conversation features become more important, but still the lexical features are the most informative; see Figure 4. In all cases relying on lexical information (see Figures 3 and 4), the features extracted by using string kernels are, as a whole, the most informative features with Gini values of 0.41 and 0.49, respectively. More discussions of the results in Table 5 can be found in the last part of Section 4.

Finally, we note that the rather low F1 scores for the task of detecting semantic links (especially compared to other NLP tasks) are mainly due to the intricate nature of the task. Solving the task of detecting semantic links in chats implies properly disentangling discourse structure, both at individual level and when multiple participants are involved throughout an evolving conversation.

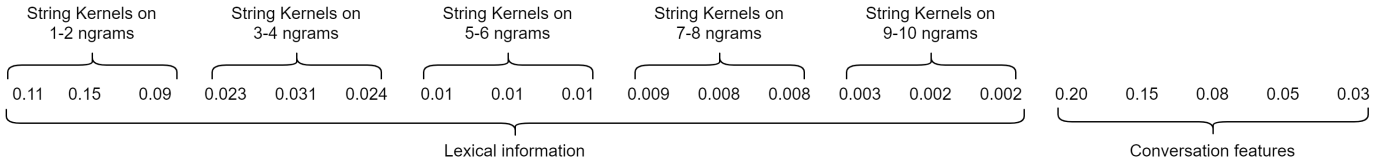

Figure 4. Gini feature importance for random forest with string kernels replacing semantic information.

3.4. Qualitative Interpretation of the Results

In this section, we provide a qualitative interpretation of the results obtained by the proposed neural model using string kernels. In the top half of Table 6 we present two simple examples of semantic links, one as direct answer to a previously stated question, and a second one as an addition to a previous idea. The proposed model is capable of detecting when an author continues their idea in a new utterance (see lower part of Table 6).

Table 6. Example of correct semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
120		***Adrian***	***so tell me why the chat could provide collaborative learning?***
121	120	Maria	for example if we have to work on a project and everybody has a part to do we can discuss it in a conference
122		***Adrian***	***you can discuss the premises but not the whole project***
123	120	Andreea	well, you can have a certain amount of info changed
124	122	Maria	you can discuss the main ideas and if someone encounters problems he can ask for help and the team will try to help him
216		***Dan***	***are there private wikis?***
217		Ana	yes...there are wikis that need authentication...
218	216	Dan	by private i understand that only certain users can modify their content

Our model also detects distant semantic links (see Table 7) and interleaved semantic links (see Table 8). Table 9 presents two examples of complex interactions in which our system correctly detects semantic links.

Table 7. Example of correct semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
139		***Lucian***	***To CHAT with a teacher and with our colleagues which have knowledge and share it with us.***
140		Florin	yes, but I do not agree that chat's are the best way to do that
141		Lucian	For example, we could read a book about NLP but we could learn much more by CHATTING with a teacher of NLP
142		Claudia	but that chat is based on some info that we have previously read
...		...	...
145	139	Sebi	yes but the best way to share your knowledge is to make publicly with other people who wants to learn something. In chats you can do this just with the people who are online, in forums everybody can share it
62		***Alexandru***	***the blog supports itself on something our firm calls a "centeredcommunity"...the owner of the blog is the center...interacting with thecenter (artist, engineer, etc.) is something very rewarding***
63		Alexandru	and i would like to underline once again the artistic side of the blog
64		Alexandru	blog is ART
65		Raluca	you can also share files in a chat conference, in realtime
66	62	Radu	you're right... blogs have their place in community interaction

Table 8. Example of correct semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
173		***Ionut***	***as the wiki grow the maintenance becomes very hard, and it consumes a lot of human resources***
174		***Bogdan***	***many users manage that information, so the chores are distributed***
175	174	Ionut	this would be a pro for wikis
176	173	Bogdan	yes, but the wiki grows along with the users that manage that wiki
186		***Costin***	***I mean can an admin take the rights to another admin even if the firstone became admin after the second one?***
187		Ionut	who can also have the right to give admin permissions
188		***Tatar***	***wikis are good for documentation but is not a communication tool***
189	188	Ionut	so "Thumbs up!" for chat here!
190	186	Bogdan	I think if some admin does not do their job, he could lose the status

Table 9. Example of correct semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
168		***Octavian***	***it s meant to improve the quality of a web site***
169		Oana	So it;s just to inform you...
170		Octavian	yes
171	168	Tibit	there isan advantage in owning a blog...by having a somehow informal kind ofrelationship...you can get people attached to your writing...
172		Tibit	you can let them kow a part of you that cannot be shown on forums and wikis
173		Tibit	you humanise the content
174		***Octavian***	***yes, i agree with you***
175	174	Oana	I don;t
176		Tibit	why so?
177		***Oana***	***i belive that on forums, you can also show that "human" part :)***
178	177	Tibit	yes ...but you cannot build a relationship
188		***Luis***	***Let ustake the socket example. Using blog the information will be posted andthe users can asks questions and receive extra information***
189	***188***	***Cristi***	***this is a dynamic advantage of the blog***
190		Cristi	but what if there was a little error in the example?
191		***Alex***	***how are they able ask questions if they can modify the page?***
192	189	Luis	in general blog ideal for presenting products, implementation ideas , for advertising
193	191	Luis	you jst put a comment

However, the model is not perfect—in about half of the cases, it is unable to detect the correct semantic link (e.g., the best accuracy is 49.09%). Nevertheless, we must consider

that the detection of correct links is a difficult task even for human readers due to complex interactions between utterances (see examples in Table 10).

Table 10. First example of **wrong** semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
156		**Delia**	**It can also make it easier to communicate with larger groups**
157		Delia	where will be having this conversation if the chat would not exist?:P
158		*Cristian*	*it's the easiest way to solve a problem indeed, but only if the person is available*
159		Delia	the availability is prerequisitive
160		Delia	yup
161		Delia	but who is not online ourdays?:)
162	156	Marian	yes but what about when you have a problem none of your friends know how to solve
288		**Andreea**	**if kids get used to it at an early age, they would have an easier time later**
289		Razvan	blog posts, chatting, video conferences are ok
290		Razvan	I thinks it's dirrect enough
291		*Andreea*	*maybe the teacher and the students could do something fun together, like a trip :)*
292	288	Mihai	yes,this is a very important thing, every teacher should promote elearning

In some cases, the model fails to capture semantic relationships due to the simplistic way of extracting the semantic information. This shortcoming can be observed in the first example of Table 11, as the model fails to capture the relatedness of the verbs "moderate" and "administarte", although in this case it can be attributed to the misspelling of the verb *administrate*. Our model also fails to reason—the model selects as the semantic link in the example presented in the bottom half of Table 11) an utterance by the same participant who refers to herself as "you".

Table 11. Second example of **wrong** semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
242		*Alex*	*whenyou need to have a discution between multiple people and you need itstored so other people would be able to read it, you should definetlyuse a forum*
243		**Luis**	**We need someone to administarte that forum. so it will waste their time**
244		Alex	what other solution you have?
245	243	Cristi	exacly, someone has to moderate the discussions
51		*Alexandru*	*and sometimes you do :)*
52		**Raluca**	**you save a text file when you only chat or save a video file if you really need it**
53		Radu	Theidea of community is the fact that most of the members should know whatis going on. If you do not want others (ousiders) to see the inside messages, than you can create a private forum and never give it's IPaddress or DNS name to anyone :)
54	51	Raluca	i agree with you here

In other cases, utterances simply do not provide enough information (see Table 12). More complex features might help overcome these limitations. Nevertheless, some limitations are also due to the way the problem was formulated, as each utterance is analysed independently.

Table 12. Third example of **wrong** semantic link prediction (**explicit link**/*predicted link*).

Utt. ID	Ref. ID	Speaker	Content
400		*Bogdan*	*let us suppose that we use these technics in a serial mode:...so, every method corresponds to a step in clasification...*
401		Mihai	yes, we can combine the powers of every techniques and come out with a very versatile machine learning software
402		**Bogdan**	**and each step can get a result...**
403	402	Mihai	or we can choose a method depending on the nature of the problem, taking into consideration that each one fits best on a few type of problems

4. Discussion

Answer selection techniques, in conjunction with string kernels, were evaluated as a method for detecting semantic links in chat conversations. We used two datasets with related tasks (i.e., detection of semantic links as a classification task or as a regression task) to validate our approach.

The proposed neural model greatly improves upon the previous state-of-the-art for semantic link detection, boosting the exact match accuracy from 32.44% (window/time frame: 5 utterances/1 min) to 47.85% (window/time frame: 10 utterances/3 min). The neural network model outperforms previous methods for all combinations of considered frames. It is important to note that the neural network models generalize very well—e.g., performance is better for larger windows. This was not the case with any other methods presented in the first part of Table 4, where we included several strong baselines from previous works.

The addition of semantic information to our model increased its performance, but not by a large margin, especially when compared to the performance gained by adding conversation features. This highlights that an answer selection framework imposed upon the semantic link detection task has some limitations. The largest observed gain for semantic information was achieved on longer frames (e.g., window/time frame: 10 utterances/3 min, improvement from 47.85% to 49.09%), which means that semantic information becomes more helpful when discriminating between a larger number of candidates (larger window).

An interesting observation is that using BERT for extracting semantic information (last part of Table 4) does not bring significant improvements. We believe this is the case because the BERT pretrained model is trained on much longer and structurally different sentences (e.g., Wikipedia texts versus chat messages).

The results on the Linux IRC reply dataset (Table 5) are compelling. The first important observation is that the usage of string kernels improves the performance for all combinations of features (see the first two columns of Table 5 with any of the last three columns of Table 5). Similar to the first set of experiments (on the CSCL chat corpus; see Table 4), semantic information helps the model better capture direct links, but it is not critical. Using string kernels to replace semantic information (see last column of Table 5) proves to be very effective, obtaining better performance than the model without string kernels, but with semantic information (first two columns of Table 5).

Based on two different datasets, two slightly different tasks (regression and classification), and two different models, we observe the same patterns: string kernels are very effective at utterance level, while state-of-the-art semantic similarity models under-perform when used for utterance similarity. Besides higher accuracy, string kernels are also a lot faster and, if used in conjunction with a neural network on top of them, achieve state of the art results with a small number of parameters. This allows the model to be trained very fast, even on a CPU.

5. Conclusions

Computer-Supported Collaborative Learning environments have shown an increased usage, especially when it comes to problem-solving tasks, being very useful in the context of online activities imposed by the COVID-19 pandemic. For such purposes, chat environments are usually seen as a suitable technology. In addition, chats can sometimes incorporate a facility that allows participants to explicitly mark the utterance they are referring to, when writing a reply. In conversations with a higher number of participants, several discussion topics emerge and continue in parallel, which makes the conversation hard to follow.

This paper proposes a supervised approach inspired by answer selection techniques to solve the problem of detecting semantic links in chat conversations. A supervised neural model integrating string kernels, semantic and conversation-specific features was presented as an alternative to complex deep learning models, especially when smaller datasets are available for training. The neural network learnt to combine the lexical and semantic features together with other conversation-specific characteristics, like the distance

and time spent between two utterances. Results were compared to findings in the question answering field and validated the proposed solution as suitable for detecting semantic links in a small dataset, such as the collection of CSCL conversations. Our best model achieves 49.09% accuracy for exact match and 53.58% for in-turn metric. String kernels were successfully combined with semantic and conversation-specific information using neural networks, as well as other classifiers such as decision trees and random forests. State-of-the-art results (0.65 F1 score on the reply class) were also obtained using string kernels for reply detection on the Linux IRC reply dataset.

String kernels provide a fast, versatile, and easy approach for analyzing chat conversations, and should be considered as a central component to any automated analysis method that involves chat conversations, while the neural network provided relevant results for the explicit links detection task, the semantic information did not bring important additional information to the network. The experiments on the reply detection task lead to very similar results. Nonetheless, an inherent limitation of our model is generated by the answer selection framework imposed on the semantic link detection task (i.e., not considering the flow of conversation and addressing utterances independently).

The proposed method allows participants to more easily follow the flow of a discussion thread within a conversation by allowing the disambiguation of the conversation threads. As such, we provide guidance while reading entangled discussion with multiple inter-twined discussion threads. This is of great support for both educational and business-related tasks. Moreover, chat participants can obtain an overview of their involvement by having access to the inter-dependencies between their utterances and the corresponding discussion threads. Another facility might aim at limiting the mixture of too many discussion topics by providing guidance to focus only on the topics of interest. With accuracy scores slightly over 50%, the proposed method may still require human confirmation or adjustment, if the detected link is not suitable. Finally, the automated detection of semantic links can be used to model the flow of the conversation by using a graph-based approach, further supporting the understanding and analysis of the conversation's rhetorical structure.

Our method does not take into account the context in which the replies occur, which might prove to be important for detecting semantic links. Further experiments target gathering an extended corpora of conversations to further advance the chat understanding subdomain.

Author Contributions: Conceptualization, T.R., M.D. and S.T.-M.; Data curation, M.M. and G.G.-R.; Formal analysis, M.M.; Funding acquisition, T.R. and M.D.; Investigation, M.M.; Methodology, M.M., S.R. and G.G.-R.; Project administration, T.R. and M.D.; Resources, G.G.-R. and S.T.-M.; Software, M.M.; Supervision, T.R., M.D. and S.T.-M.; Validation, M.M., S.R. and G.G.-R.; Visualization, M.M.; Writing—original draft, M.M.; Writing—review & editing, S.R., T.R., M.D. and S.T.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS—UEFISCDI, project number TE 70 PN-III-P1-1.1-TE-2019-2209, "ATES—Automated Text Evaluation and Simplification" and POC-2015 P39-287 IAVPLN.

Institutional Review Board Statement: The study was conducted according to the guidelines of the Declaration of Helsinki. Both datasets (ChatLinks and Reply Annotations) used in this work are from external sources and are available freely online.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: In this work, we use two public corpora available at: https://huggingface.co/datasets/readerbench/ChatLinks (accessed on 10 November 2021) and http://shikib.com/td_annotations (accessed on 10 November 2021).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Cinelli, M.; Quattrociocchi, W.; Galeazzi, A.; Valensise, C.M.; Brugnoli, E.; Schmidt, A.L.; Zola, P.; Zollo, F.; Scala, A. The COVID-19 social media infodemic. *Sci. Rep.* **2020**, *10*, 16598.
2. Alalwan, N.; Al-Rahmi, W.M.; Alfarraj, O.; Alzahrani, A.; Yahaya, N.; Al-Rahmi, A.M. Integrated three theories to develop a model of factors affecting students' academic performance in higher education. *IEEE Access* **2019**, *7*, 98725–98742. [CrossRef]
3. Dutt, A.; Ismail, M.A.; Herawan, T. A systematic review on educational data mining. *IEEE Access* **2017**, *5*, 15991–16005. [CrossRef]
4. Chen, L.; Chen, P.; Lin, Z. Artificial intelligence in education: A review. *IEEE Access* **2020**, *8*, 75264–75278. [CrossRef]
5. Rashid, N.A.; Nasir, T.M.; Lias, S.; Sulaiman, N.; Murat, Z.H.; Abdul Kadir, R.S.S. Learners' Learning Style classification related to IQ and Stress based on EEG. *Procedia Soc. Behav. Sci.* **2011**, *29*, 1061–1070. [CrossRef]
6. Rashid, N.A.; Taib, M.N.; Lias, S.; Bin Sulaiman, N.; Murat, Z.H.; Abdul Kadir, R.S.S. EEG theta and alpha asymmetry analysis of neuroticism-bound learning style. In Proceedings of the 2011 3rd International Congress on Engineering Education: Rethinking Engineering Education, The Way Forward, ICEED 2011, Kuala Lumpur, Malaysia, 7–8 December 2011; pp. 71–75. [CrossRef]
7. Anaya, A.R.; Boticario, J.G. Clustering learners according to their collaboration. In Proceedings of the 2009 13th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2009, Santiago, Chile, 22–24 April 2009; pp. 540–545. [CrossRef]
8. Rus, V.; D'Mello, S.; Hu, X.; Graesser, A.C. Recent advances in conversational intelligent tutoring systems. *AI Mag.* **2013**, *34*, 42–54. [CrossRef]
9. Zhang, H.; Magooda, A.; Litman, D.; Correnti, R.; Wang, E.; Matsmura, L.C.; Howe, E.; Quintana, R. eRevise: Using natural language processing to provide formative feedback on text evidence usage in student writing. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 9619–9625.
10. Hamdi, M.S. MASACAD: A multi-agent approach to information customization for the purpose of academic advising of students. *Appl. Soft Comput. J.* **2007**, *7*, 746–771. [CrossRef]
11. Kolb, D.A. *Experiential Learning: Experience as the Source of Learning and Development*; Prentice Hall: Englewood Cliffs, NJ, USA, 1984.
12. Dascalu, M.D.; Ruseti, S.; Dascalu, M.; McNamara, D.S.; Carabas, M.; Rebedea, T.; Trausan-Matu, S. Before and during COVID-19: A Cohesion Network Analysis of Students' Online Participation in Moodle Courses. *Comput. Hum. Behav.* **2021**, *121*, 106780. [CrossRef]
13. Stahl, G. *Group Cognition. Computer Support for Building Collaborative Knowledge*; MIT Press: Cambridge, MA, USA, 2006.
14. Blanco, M.; Gonzalez, C.; Sanchez-Lite, A.; Sebastian, M.A. A practical evaluation of a collaborative learning method for engineering project subjects. *IEEE Access* **2017**, *5*, 19363–19372. [CrossRef]
15. Trausan-Matu, S. Computer support for creativity in small groups using chats. *Ann. Acad. Rom. Sci. Ser. Sci. Technol. Inf.* **2010**, *3*, 81–90.
16. Trausan-Matu, S. The polyphonic model of collaborative learning. In *The Routledge International Handbook of Research on Dialogic Education*; Mercer, N., Wegerif, R., Major, L., Eds.; Routledge: London, UK, 2019; pp. 454–468.
17. Holmer, T.; Kienle, A.; Wessner, M. Explicit referencing in learning chats: Needs and acceptance. *Innov. Approaches Learn. Knowl. Shar. Proc.* **2006**, *4227*, 170–184. [CrossRef]
18. Masala, M.; Ruseti, S.; Rebedea, T. Sentence selection with neural networks using string kernels. *Procedia Comput. Sci.* **2017**, *112*, 1774–1782. [CrossRef]
19. Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 2383–2392.
20. Shen, G.; Yang, Y.; Deng, Z.H. Inter-Weighted Alignment Network for Sentence Pair Modeling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 9–11 September 2017; pp. 1179–1189.
21. dos Santos, C.; Tan, M.; Xiang, B.; Zhou, B. Attentive Pooling Networks. *CoRR* **2016**, *2*, 4.
22. Yu, L.; Hermann, K.M.; Blunsom, P.; Pulman, S. Deep learning for answer sentence selection. *arXiv* **2014**, arXiv:1412.1632.
23. Masala, M.; Ruseti, S.; Gutu-Robu, G.; Rebedea, T.; Dascalu, M.; Trausan-Matu, S. Help Me Understand This Conversation: Methods of Identifying Implicit Links Between CSCL Contributions. In Proceedings of the European Conference on Technology Enhanced Learning, Leeds, UK, 3–6 September 2018; Springer: Cham, Switzerland, 2018; pp. 482–496. [CrossRef]
24. Masala, M.; Ruseti, S.; Gutu-Robu, G.; Rebedea, T.; Dascalu, M.; Trausan-Matu, S. Identifying implicit links in CSCL chats using string kernels and neural networks. In Proceedings of the International Conference on Artificial Intelligence in Education, London, UK, 27–30 June 2018; Springer: Cham, Switzerland, 2018; pp. 204–208. [CrossRef]
25. Gutu, G.; Dascalu, M.; Rebedea, T.; Trausan-Matu, S. Time and Semantic Similarity—What is the Best Alternative to Capture Implicit Links in CSCL Conversations? In Proceedings of the 12th International Conference on Computer Supported Collaborative Learning (CSCL), Philadelphia, PA, USA, 18–22 June 2017; pp. 223–230.
26. Gutu, G.; Dascalu, M.; Ruseti, S.; Rebedea, T.; Trausan-Matu, S. Unlocking the Power of Word2Vec for Identifying Implicit Links. In Proceedings of the IEEE 17th International Conference on Advanced Learning Technologies, ICALT 2017, Timisoara, Romania, 3–7 July 2017; pp. 199–200. [CrossRef]
27. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
28. Wu, Z.; Palmer, M. Verb Semantics and Lexical Selection. In Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, Las Cruces, NM, USA, 27–30 June 1994; pp. 133–138. [CrossRef]

29. Leacock, C.; Chodorow, M. Combining Local Context and WordNet Similarity for Word Sense Identification. In *WordNet: An Electronic Lexical Database*; Fellbaum, C., Ed.; MIT Press: Cambridge MA, USA, 1998; pp. 265–283.
30. Landauer, T.K.; Dumais, S.T. A solution to Plato's problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychol. Rev.* **1997**, *104*, 211–240. [CrossRef]
31. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representation in Vector Space. In Proceedings of the Workshop at ICLR, Scottsdale, AZ, USA, 2–4 May 2013.
32. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2013; pp. 1–9.
33. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543. [CrossRef]
34. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
35. Lodhi, H.; Saunders, C.; Shawe-Taylor, J.; Cristianini, N.; Watkins, C. Text Classification using String Kernels. *J. Mach. Learn. Res.* **2002**, *2*, 419–444. [CrossRef]
36. Ionescu, R.T.; Popescu, M.; Cahill, A. Can characters reveal your native language? A language-independent approach to native language identification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1363–1373.
37. Ionescu, R.T.; Popescu, M.; Cahill, A. String Kernels for Native Language Identification: Insights from Behind the Curtains. *Comput. Linguist.* **2016**, *42*, 491–525.
38. Gönen, M.; Alpaydın, E. Multiple Kernel Learning Algorithms. *J. Mach. Learn. Res.* **2011**, *12*, 2211–2268.
39. Beck, D.; Cohn, T. Learning Kernels over Strings using Gaussian Processes. In Proceedings of the Eighth International Joint Conference on Natural Language Processing, Taipei, Taiwan, 27 November–1 December 2017; Volume 2, pp. 67–73.
40. Bachrach, Y.; Zukov-Gregoric, A.; Coope, S.; Tovell, E.; Maksak, B.; Rodriguez, J.; McMurtie, C. An Attention Mechanism for Answer Selection Using a Combined Global and Local View. *arXiv* **2017**, arXiv:1707.01378.
41. Tan, M.; dos Santos, C.; Xiang, B.; Zhou, B. Improved Representation Learning for Question Answer Matching. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 464–473.
42. Graves, A.; Schmidhuber, J. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef] [PubMed]
43. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
44. Luong, M.T.; Pham, H.; Manning, C.D. Effective approaches to attention-based neural machine translation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1412–1421.
45. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
46. Wang, S.; Jiang, J. A Compare-Aggregate Model for Matching Text Sequences. *arXiv* **2016**, arXiv:1611.01747.
47. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 5998–6008. [CrossRef]
48. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
49. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. *Improving Language Understanding by Generative Pre-Training*; Technical Report; OpenAI: San Francisco, CA, USA, 2018.
50. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. *Language Models Are Unsupervised Multitask Learners*; Technical Report; OpenAI: San Francisco, CA, USA, 2019.
51. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
52. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 5754–5764.
53. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020.
54. Trausan-Matu, S.; Rebedea, T. A polyphonic model and system for inter-animation analysis in chat conversations with multiple participants. In Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics, Iasi, Romania, 21–27 March 2010; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6008, pp. 354–363. [CrossRef]
55. Elsner, M.; Charniak, E. Disentangling chat. *Comput. Linguist.* **2010**, *36*, 389–409. [CrossRef]

56. Jiang, J.Y.; Chen, F.; Chen, Y.Y.; Wang, W. Learning to disentangle interleaved conversational threads with a siamese hierarchical network and similarity ranking. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 1812–1822.
57. Liu, H.; Shi, Z.; Gu, J.C.; Liu, Q.; Wei, S.; Zhu, X. End-to-End Transition-Based Online Dialogue Disentanglement. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, Yokohama, Japan, 7–15 January 2020; pp. 3868–3874.
58. Li, T.; Gu, J.C.; Zhu, X.; Liu, Q.; Ling, Z.H.; Su, Z.; Wei, S. DialBERT: A Hierarchical Pre-Trained Model for Conversation Disentanglement. *arXiv* **2020**, arXiv:2004.03760.
59. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
60. Mehri, S.; Carenini, G. Chat Disentanglement: Identifying Semantic Reply Relationships with Random Forests and Recurrent Neural Networks. In Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP2017), Taipei, Taiwan, 27 November–1 December 2017; pp. 615–623.
61. Searle, J.R. *Speech Acts*; Cambridge University Press: Cambridge, UK, 1969. [CrossRef]
62. Moldovan, C.; Rus, V.; Graesser, A.C. Automated Speech Act Classification for Online Chat. *MAICS* **2011**, *710*, 23–29.
63. Rus, V.; Maharjan, N.; Tamang, L.J.; Yudelson, M.; Berman, S.R.; Fancsali, S.E.; Ritter, S. An Analysis of Human Tutors' Actions in Tutorial Dialogues. In Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS 2017), Marco Island, FL, USA, 22–24 May 2017; pp. 122–127.
64. Stahl, G. *Studying Virtual Math Teams*; Springer Science & Business Media: New York, NY, USA, 2009.
65. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015. [CrossRef]
66. Hu, B.; Lu, Z. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; pp. 1–9.
67. Lowe, R.; Pow, N.; Serban, I.; Pineau, J. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv* **2015**, arXiv:1506.08909.
68. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]