Improving Text Accessibility using Text Simplification and Conversation Disentanglement

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computational Linguistics
by Research

by

KV Aditya Srivatsa
2018114018
k.v.aditya@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
July, 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Improving Text Accessibility using Text Simplification and Conversation Disentanglement" by KV Aditya Srivatsa, has been carried out under my supervision and is not submitted elsewhere for a degree.

July 8, 2023	Adviser: Prof. Manish Shrivastava

To my grandparents

Acknowledgments

I would like to express my sincere gratitude to my advisor, Prof. Manish Shrivastava, for guiding me through my first steps in research and offering their extensive subject knowledge and guidance. They have been there for me even when a research direction did not pan out, and their constant support gave me the confidence to continue working toward research problems that truly intrigue me. I also thank them for their countless hours reviewing and editing my work and helping me convey my ideas better.

Next, I would like to thank the professors in the LTRC labs for taking me through courses that piqued my interest in Linguistics, NLP, and Information Retrieval, specifically Prof. Radhika Mamidi, Prof. Dipti Misra Sharma, Prof. Vasudev Verma, and many others who helped me inculcate the knowledge and perspective required for my research.

I would also like to thank my friends who helped me develop the ideas and skills for my research. Mukund Choudhary played an important role in helping me formulate research problems, plan and execute key aspects of my projects, and provide thought-provoking perspectives. Ishan Sanjeev Upadhyay, Monil Gokani, and Nomaan Qureshi helped me gain invaluable experience with paper writing, applying machine learning techniques, and identifying novel directions in my research. Prashant Kodali helped me manage various logistical aspects of my research papers and thesis. Other friends and lab-mates, including Bharathi Ramana Joshi, Rishav Kundu, Tathagata Raha, and many others, were a constant source of discussion, support, and positivity pivotal in keeping me driven toward my work.

I would like to thank my father, mother, and brother for their irreplaceable guidance and constant faith in me. Finally, I would like to thank my grandparents, who stood by me throughout my research journey, especially during the trying times of the pandemic, and encouraged me to continue striving toward my goals.

Abstract

Accessibility to text for machines i.e. for downstream tasks, has been improved greatly using mainstream NLP tasks such as Machine Translation and Summarization. In this dissertation, however, we delve into two lesser-explored research areas: Text Simplification (reducing the linguistic complexity of text) and Conversation Disentanglement (extracting individual conversation threads from multi-party dialogues).

We pursue text simplification by analyzing the claim of utilizing control tokens in input text to produce simplified versions with precise variations along key text attributes [1]. The effect of control tokens on simplifications was isolated individually and in particular combinations through dedicated models. We show that though control tokens can help mold the outputs closer to the target simplifications, they often cause undesired interactions between control tokens and text attributes as well as within control tokens. This sometimes results in outputs being trivial source text variants that adhere to the control token constraints but do not furnish true simplifications. We offer methods to curb some of these trivial outputs e.g. masking named-entities to shield them from unwanted replacements and to curb the extent of data sparsity using pre-trained input embeddings. Our modifications produce fewer trivial outputs and show better resilience to unseen contexts.

As part of our work in conversation disentanglement, we explore the most widely used expert-annotated resource in the field: Ubuntu-IRC (IRC) dataset [2]. We question IRC's utility for training generalizable models for other domains which currently lack similar large-scale, high-quality datasets, e.g., meeting transcripts & movie dialogues. We address this concern by analyzing IRC for potential sources of platform-based specificity. We find that using direct mentions (explicit references to a message's recipient) in IRC is one such instance of platform specificity that occurs abundantly in the dataset. To measure its impact on model performance, we create Ubuntu-IRC-Hard – a variant of IRC without direct mentions. We show that the performance of past approaches (and proposed baselines) degrades significantly without direct mentions. An in-depth analysis of models is conducted using qualitative metrics based on dialogue properties which reveal brittleness and variations among models across properties. We introduce two methods that leverage these insights to alleviate this brittleness: (i) A weighted-loss formulation to better represent less frequent sub-ranges of dialogue properties in the data, and (ii) An informed-ensembling technique that determines a subset of models best suited for each property sub-range. Both methods surpass SOTA [3] performance and score more consistently across dialogue properties.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Preamble

The amount of text available in the World on various platforms is increasing rapidly through books, websites, social media apps, official documents, etc. However, not all text is equally easy to access, specifically for machines to process and utilize for downstream tasks. Improving text accessibility for machines spans across different axes e.g. translating text to a familiar language, distilling to a manageable length, or simplifying to aid comprehension.

Simplification of any form of text **involves identifying any relevant underlying structure and making it apparent**. For instance, for sentence level simplification, the underlying structure constitutes the syntactic tree, detailing the different clauses in the sentence. Upon segmenting the sentence along these clauses, the resultant chunks of text are more suited for interpretation. Alternatively, simplification may also involve the elimination of noise. In sentence simplification, the use of elitist words or any purposeful obfuscation act as noise and impedes comprehension. Therefore, for downstream tasks, sentence level simplification often serves as a text regularization step.

Similarly, consider an online chatting platform like the **Internet Relay Chat** which records incoming messages of a multi-party conversation in a linear fashion. Here, the complexity arises for a downstream system, from the fact that multiple users are conversing with each other, wherein, a user may be actively involved in multiple disjointed conversation threads. Without explicit information regarding which messages belong in a thread, or which user is being addressed in an utterance, crucial information may be lost for any downstream discourse comprehension task. The underlying structure here of course is the thread structure, which needs to be made apparent. Alternatively, from a simplification-as-denoising perspective, such a multi-party conversation is posed as a noisy-room setting. In the process of comprehending each thread, messages and active users belonging to other threads act as noise, each thread needs to be disentangled (conversation disentanglement).

Despite their differences, both **sentence simplification** and **conversation disentanglement** aim to make a discourse more comprehensible. Text simplification aids sentence-level comprehension, while conversation disentanglement aids in isolating individual discourses in the first place. Both techniques have the potential to make text more accessible for further processing or inference.

For instance, text simplification can be used to improve the performance of downstream NLP tasks, such as machine translation and summarization. Similarly, conversation disentanglement can be applied to improve the performance of discourse understanding systems. Therefore, through this thesis, we record our contributions towards sentence simplification and conversation disentanglement, which can make text better accessible to a plethora of tasks.

1.2 Scope of the Thesis

This section briefly describes the two tasks mentioned earlier and identifies important areas for further research. We then outline the specific directions explored in this thesis and our contributions to these areas.

1.2.1 Text Simplification

Text Simplification (or Sentence Simplification if specified) is the task of making text easier to comprehend i.e. improving its readability. This may be done by modifying various attributes of the text, such as using *simpler* words (more generic terms in place of technical or domain-specific terms), breaking down extremely long sentences into smaller, more manageable chunks, and using more familiar syntactic structures (e.g. omitting the use of highly nested constructions). These transformations can make understanding text easier for young or second-language learners or people with cognitive impairments. As simplification of text is closely associated with human familiarity with certain vocabulary and writing styles and remains to be a fairly subjective task (i.e. no single text is the true simplified form), the high-quality datasets for text simplification contain multiple simplified references for every complex source text.

Text simplification has largely been modeled as a **monolingual machine translation task**, with initial works making use of statistical machine translation [4] and later shifting to deep contextualized Seq2Seq models. However, as many instances of simplification involve modifying only a few words in the source text, the expected simplifications often closely resemble the source. This creates a significant bias for most Seq2Seq models to trivially predict the complex source as the output. To address this, the **ACCESS** model [1] proposes the use of control tokens to allow translation models to better encode individual text transformations such as shortening of text, a certain degree of paraphrasing, and lowering of the average lexical and syntactic complexity. Their method is shown to outperform past approaches on the WikiLarge corpus. With this approach in mind, we enumerate a few directions of research worth exploring:

1. Creating more large-scale aligned datasets, incorporating detailed information on the textual transformations relevant to simplification e.g. lexical and syntactic complexity.

2. Analyzing and improving the control over fine-grained aspects of the simplified text produced.

3. Developing modeling techniques that represent sentence-level simplification better than the widely used Seq2Seq models e.g. edit-based methods.

This thesis focuses on direction 2. We assess the ACCESS model [1] to better understand the role of control tokens in tuning the simplifications produced. We identify points where the claimed effect of the control tokens on their respective attributes breaks down and propose methods to curb this mismatch.

1.2.2 Conversation Disentanglement

Dialogue (or Conversation) Disentanglement, i.e. the partitioning of utterances in a multi-party dialogue by their respective conversation threads, is a crucial sub-task for modeling end-to-end dialogue systems as well as providing a better interface to chat-room conversations e.g. in WhatsApp groups. However, the manually supervised data required for this task requires lengthy and granular reviews, with utterances in a chat set apart by hundreds of potentially linked messages. Additionally, the task remains subjective to a degree, with annotators often disagreeing upon exact thread partitions. Although there have been efforts to extract thread partitioning without manual annotation, such as utilizing the reply structure of platforms with built-in thread conventions e.g. Reddit, or artificially entangling single or two-party conversations, these attempts fail to produce organic distributions.

The primary dataset for the scope of our thesis contributes significantly to this end. The **Ubuntu-IRC** dataset [2] contains over 70,000 messages collected from the `#Ubuntu` channel from the Internet Relay Chat, with each message annotated for their reply-to relation. With additional dialogue context for each session (to avoid trivial thread initiations) and adjudicated annotations, the IRC dataset is presently the **only large-scale expert-annotated dataset for dialogue disentanglement**.

Therefore, though it is necessary to develop high-quality data across domains other than IRC, it is equally imperative to maximize the usability of the Ubuntu-IRC dataset for the analysis of dialogue platforms and training & evaluating disentanglement models until an ample variety of datasets is made available. Taking this into account, we provide a consolidated list of research directions worthy of exploration for furthering our understanding and ability in conversation disentanglement:

1. Creating datasets of similar scale and quality for other conversational domains e.g. meeting transcripts and movie dialogues.

2. Re-evaluating the feasibility of existing datasets for representing the task of chat disentanglement.

3. Making disentanglement modeling techniques robust across different kinds of multi-party conversations (within & across dialogue platforms).

4. Developing methods to leverage existing datasets to adapt to poorly represented or unseen dialogue domains.

In this dissertation, we primarily pursue directions 2 and 3. We carry out an extensive analysis of the primary dataset for Ubuntu-IRC along key dialogue properties (see Section 4.1) to identify platform-specific features which may affect the performance of current modeling techniques significantly. We also propose methods to (i) curb the poor representation of low-frequency sub-ranges of important dialogue properties and (ii) optimally combine the capabilities of different disentanglement models based on the relevant dialogue context.

1.3 Research Problems Addressed

In this thesis, we address the following computational research problems in order to progress toward some of the objectives described in the above section.

T1 *To determine the impact of control tokens on the characteristics of sentence simplifications and gauge whether they hold the expected control over their respective attributes.* The ACCESS model [1] prepends control tokens to the input complex text to a Seq2Seq Transformer, where the expected output is the simplified reference text. Each control token quantifies the variation in one textual attribute between the complex and simple text. These include variation in sequence length, paraphrasing of the original text, and change in lexical and syntactic complexity. To capture these changes, the tokens encode respective proxies of these attributes. The intended purpose of these tokens is to inform the model of these underlying transformations during simplification and later tweak them as per the desired form of simplification. We aim to analyze the extent to which these proxies represent and thereby help control the attributes of the text.

T2 *To develop methods to improve the intended effect of control tokens on the output simplifications.* The reported scores by the ACCESS models indicate that the use of control tokens helps improve performance over using a plain Transformer model. Despite this, the resultant effect of control tokens on the text may not be along the intended attributes. We, therefore, aim to counteract instances wherein the effect of the control tokens differs from the expected. We thus propose methods such as NER masking (see Section 2.6.1) and using pre-trained embeddings (see Section 2.6.2) to tend to some of such cases.

T3 *To analyze the distribution of each feature in key dialogue datasets towards model performance and identify potential platform-specific features.* Popular disentanglement datasets (like Ubuntu IRC) offer meta information for each message, the sender username, and the message body. These may be interface-integrated features like message timestamps or unofficial norms followed by the community, such as adding an explicit mention of the intended recipient of a message at the beginning of the message body (direct mention). In this thesis, we perform a detailed analysis of all such features to identify trends among their independent as well as combined distributions.

T4 *To gauge the role of platform-specific features in the performance of past approaches toward disentanglement.* Platform-specific features may boost the performance of models along with evaluation metrics. Still, they may impede the model from learning important patterns within the message text as well as inter-user interactions. To understand how much recent disentanglement approaches rely on one such feature, i.e., direct mentions in the Ubuntu-IRC dataset, we create a variant without direct mentions and record the scores of several past methods on popular metrics.

T5 *To devise a more representative evaluation method aimed at comparing model performance toward disentanglement.* Existing metrics for chat disentanglement are borrowed from the task of group partitioning, wherein the group represents all the messages of a session, and each sub-group (or partition) represents a distinct non-overlapping thread of messages. However, the metrics in use regard all kinds of threads with the same weightage in the final scalar score. For datasets with non-uniform distributions across dialogue properties, this hides the variation in model performance across the ranges of each dialogue property. To this end, we propose an evaluation strategy that compares the thread correctness (see 5.2.2) of models across the range of key dialogue properties.

T6 *To identify methods to improve model performance along specific dialogue characteristics or combine multiple models with varying capabilities for better overall performance.* We find that different models (along architecture and training methodology) possess different capabilities across dialogue properties. We offer methods to (1) Optimize model performance in cases where the disparity is due to a disproportionate distribution of dialogue samples across the range of a dialogue property and (2) Selectively apply the optimal model from a large pool of models, subject to the required capabilities in a dialogue context. Past ensembling methods such as [5] aim to combine models toward a similar goal but lack the ability to dynamically select the subset of models deemed ideal for a particular instance.

1.4 Overview

This section describes some of the key observations from our investigations into both tasks. These observations also help motivate subsequent attempts to address potential issues or leverage observed patterns to improve modeling techniques.

1.4.1 Text Simplification

1.4.1.1 Poor Attribute Control

We train and compare four Transformer models, each equipped with a single control token, to assess their individual effect on their corresponding attribute as well as other attributes. We find that tokens such as NbChars (for compression ratio) and LevSim (for the extent of paraphrasing) control their attributes fairly well. However, other tokens i.e. WordRank (for lexical complexity) and DepTreeDepth (for syntactic complexity), fail to represent their corresponding attributes as well as their proxy quantities.

We also find that a single control token can and often affects other attributes. For instance, lowering the value for NbChars effect the Levenshtein similarity as well as the dependency tree depth of the text. We later concluded this was so, as the model often produced outputs with trivial transformations applied to the source text. E.g. The model produced outputs that were truncated versions of the source text. This trivially conforms to a low-set compression ratio, Levenshtein similarity, and dependency depth.

Another interesting instance of poor attribute control leading to unwanted modification in the text is the dropping of named entities. This helps trivially reduce the *estimated* lexical complexity of the text, as the vocabulary word-rank was used as its proxy. High-rank proper nouns were replaced with low-rank pronouns to reduce the average word-rank of the text while retaining other complex content words.

1.4.1.2 Induced Data Sparsity

The training procedure prescribed with the ACCESS model requires the Transformer model to be trained from scratch. Additionally, the vocabulary was updated with a large number of newly initialized tokens pertaining to the entire range of the control tokens used in the text. Combining these two steps induces heavy data sparsity i.e. there now exists a large number of randomly initialized token embeddings that need to be learned alongside learning the transformations for simplification, solely using the dedicated aligned dataset.

1.4.2 Conversation Disentanglement

1.4.2.1 Lack of Generalizability

The Ubuntu-IRC dataset is currently the only large-scale expert-annotated resource for training and evaluating complex disentanglement systems. Hence, in this thesis, we closely assess the utility of Ubuntu-IRC in developing models applicable to domains beyond online chat forums like IRC, such as meeting transcripts & movie dialogues.

We address this by analyzing the Ubuntu-IRC dataset along several dialogue properties to identify potential sources of specificity. If left unchecked, these may hamper the generalizability of models trained on this data. We find that **direct mentions** (i.e., explicit references to the recipient user at the start of the message body) contribute to the specificity of IRC and occur in over 44% of messages in IRC. These mentions are an informal norm within online forums like IRC but may not be organic to other dialogue domains. Therefore, we create **Ubuntu-IRC-Hard**, a variant of the original IRC dataset with all instances of direct mentions removed. We report the performance of several past methods and

```
[01:17] <arun> somebody help me to mount...
[01:17] <louisdk> Kitt3n: Some core pack...
[01:17] <Kitt3n> louisdk, EOL means EOL,...
[01:17] <dr_willis> read the urls the bo...
[01:17] <arun> somebody help me to mount...
[01:17] <dr_willis> !mount | arun
[01:17] <ubottu> arun: mount is used to ...
[01:17] <devmedoo> ok dr_willis
[01:19] <dr_willis> devmedoo: you did ve...
[01:19] <arun> i can access all of my ha...
[01:20] <Kitt3n> arun, if you can access...
```

Figure 1.1: Example of direct-mentioning in IRC. usernames in bold are instances of direct-mentions, pointing to the intended recipient of the message.

a strong set of proposed baselines on both dataset variants. We find that direct mentions significantly contribute to the performance of existing approaches. Figure 1.2 shows this in the drop in performance of the SOTA model [3] as the proportion of direct mentions in Ubuntu-IRC is reduced.

1.4.2.2 Model Brittleness

In Section 4.1, we define *dialogue properties* as the set of properties emerging from the conversation in a chat forum where multiple dialogues, each with a varying number of participants, run simultaneously. Scores on existing thread-partitioning metrics do not always capture the variation in model performance across different dialogue properties. We conduct an analysis of the models using qualitative metrics based on a few key dialogue properties to identify points of brittleness among models, aiding our investigation into their generalizability.

Of these dialogue properties, we find that *antecedent distance* (message-distance between a message and its *reply-to* parent) is often a failure point for most models. This is partly due to a poor representation of longer antecedent distances in the IRC dataset.

1.4.2.3 Varying Capabilities

Additionally, we find that models show varied capabilities across most dialogue properties. Models rated superior by current partitioning metrics do not always surpass other approaches for the entire range of a dialogue property.

We attempt to increase model performance along longer antecedent distances by utilizing a weighted-loss formulation to inversely scale the loss for instances with a lower representation in the data. This provides a heightened weightage to examples with longer antecedent distances in the data, resulting in a better model performance than the SOTA along longer distances. We leverage the variations in model performances across a dialogue property by applying an informed-ensembling technique. From a

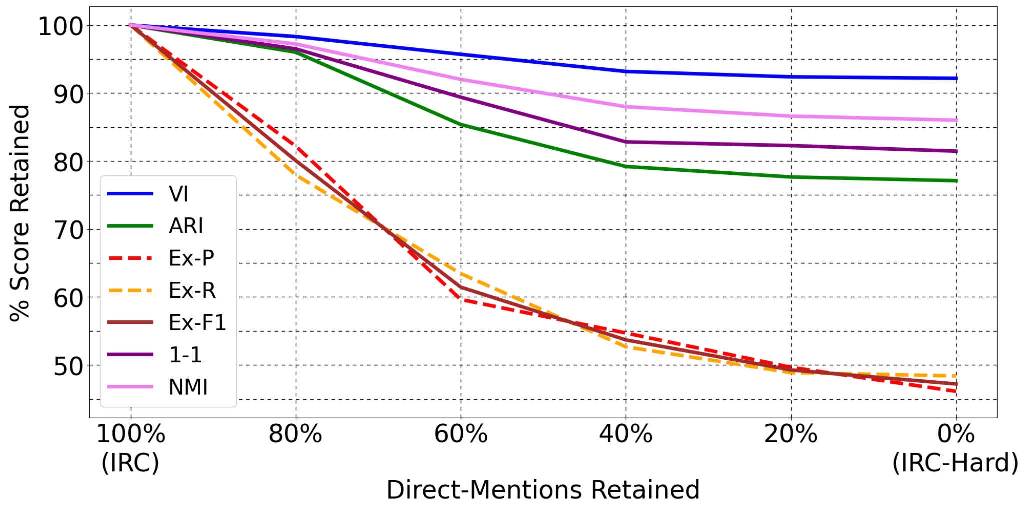


Figure 1.2: Drop in SOTA model performance with a decreasing proportion of direct mentions

diverse pool of models, the ensemble dynamically samples a subset of models best suited for the specific sub-range of the relevant dialogue property (instead of always using the entire pool).

1.5 Thesis Layout

C1 This is the **introductory chapter**, wherein we define the scope of the investigations and experiments we conduct towards analyzing disentanglement datasets and improving both the performance as well as the evaluation of models. We enlist the specific research problems addressed in this thesis, the motivation behind them, and a quick summary of our approaches that will follow in later chapters.

C2 In this chapter, we address the first area of research towards text accessibility i.e. **text simplification**. We discuss the scope and purpose of the task and provide a background into the past modeling approaches used for simplification. We discuss the ACCESS model [1], which claims to utilize control tokens to tune the simplifications produced along specific text transformations such as text compression, amount of paraphrasing, and change in the lexical & syntactic complexity of the source text. We analyze this claim by carrying out ablation studies and measure the effect of the control tokens on their attributes, and show that this claim breaks down in several cases. We conclude by offering some methods to tend to this gap, and improve the overall scores of the SOTA model.

C3 Here, pivoting to the second area of research towards text accessibility, we begin by formalizing the definition of **dialogue disentanglement** and the constraints within which the task will be discussed. We then describe the datasets created for disentanglement, with emphasis on Ubuntu-

IRC, the primary dataset in discussion for our thesis. This follows a record of the various modeling techniques along varying features and evolving architectures. Here, we again focus on S4DD [3], the current SOTA work in chat disentanglement using Ubuntu-IRC. Finally, we summarize the popular evaluation metrics used by past work.

C4 In this chapter, we introduce the term **dialogue properties**, i.e., the properties of a multi-party chat room that emerge by virtue of how dialogues between multiple speakers, spanning multiple threads, are recorded in a linear fashion temporally. We list several such properties and go on to discuss the distributions of some key dialogue properties in the context of the Ubuntu-IRC dataset. We then delve deeper into the role of a commonly occurring feature of IRC i.e. direct mentions, along a few dialogue properties. We conclude by motivating the need to investigate further the impact of this platform-specific feature on the performance of past models.

C5 In this chapter, we pursue the investigation of the **impact of direct mentions**. We construct a variant of Ubuntu-IRC with direct mentions removed (Ubuntu-IRC-Hard) and describe the steps to reproduce the dataset in detail. As part of the experiments on this dataset, in addition to past models, we propose a list of baseline models which capture each of the pivotal modules shown to be of import in past approaches. We describe the implementation details for all training runs as well as the evaluation methodologies applied. We record the performance of past and baseline models and discuss their results along popular partitioning metrics as well as the newly proposed qualitative metrics along key dialogue properties.

C6 This chapter proposes methods to **leverage variations in model capability to improve overall performance**. We demonstrate using antecedent distance that using a soft weighted loss formulation during training can help models capture poorly represented sub-ranges of a dialogue property better (see 6.2). We also offer an informed ensembling method to dynamically select a subset of models best suited for a given dialogue instance from a large pool of models. Using both methods, we report improvement in scores over the SOTA model across partitioning metrics and the ranges of several dialogue properties.

C7 We **conclude** this thesis by summarizing our analyses, subsequent experiments, and the resultant insights. We also offer several directions in which this work can be further extended in the future.

Chapter 2

Text Simplification

2.1 Introduction

Sentence Simplification aims to improve the readability of text by lowering its linguistic complexity while preserving all the relevant information conveyed by the initial text. As part of our larger goal of aiding the accessibility to text, simplification helps to regularize text along the axes of lexical distribution and syntactic constructions. It helps break up large text into smaller, more manageable chunks. This makes the text usable for downstream tasks by bringing the writing style, average sentence length, and overall vocabulary closer to distributions more suited for other NLP tasks such as machine translation, summarization, etc.

This chapter analyzes the present SOTA method used for sentence simplification. The model i.e. ACCESS [1], comprises a Transformer model [6], which is trained on a dedicated sentence simplification dataset along with control tokens prepended to the complex source sentences. These control tokens are intended to encode transformations along specific textual properties such as length compression, the extent of paraphrase, and lexical & syntactic complexity as attributes to gauge the transformations between complex and simple sentence pairs. The analysis is aimed at gauging the validity of the claim that the proposed tokens, in fact, encode and thereby help control their respective attributes and to determine any potential sources of errors in the simplified outputs.

Our analysis shows that though control tokens help improve the model's overall performance, varying their values does not always result in a corresponding effect on their respective attributes. In fact, the model often produces outputs that manage to adhere to the specified control token constraints using a trivial modification to the complex source text. This includes truncating the source text or replacing named entities with pronouns. We also find that many of the target attributes are interrelated, therefore, cannot be tweaked independently. Attempts to do so often result in incoherent outputs.

We finally discuss a few methods to address the pitfalls of the SOTA model [1]. First, we propose masking all named-entities (NE) in the text (source and target) and saving a mapping for later use. The NE masking protects low-frequency proper nouns from getting replaced or dropped to lower the lexical complexity of the text and also helps optimize the vocabulary size. Second, we propose the idea of using pretrained embeddings or pretraining the Transformer model instead of training on the sentence simplification data from scratch. This may help develop more representative token embeddings. It will populate the vocabulary with a more rich distribution of complex and simple terms, which may aid in generating outputs more dissimilar from the source and better suited to the desired control tokens. Our code has been made publicly available. [1]

2.2 Background

One prominent approach to sentence simplification is to frame it as a monolingual translation task. This implies training MT architectures over complex-simple sentence pairs, either aligned manually [7, 4] or automatically [8, 9] using large complex-simple repository pairs such as the English Wikipedia and the Simple English Wikipedia.

Another approach to sentence simplification involves controlled text generation, where specific criteria or constraints are imposed on the generated output. [10] developed a controlled text simplification system that allowed users to specify the desired level of simplification, enabling the generation of sentences that catered to individual needs. A similar work [11] aims to control attributes like length, paraphrasing, lexical complexity, and style.

Controlled paraphrasing is another technique employed in sentence simplification. [12] proposed a reinforcement learning-based approach for controlled sentence simplification paraphrasing, where a user-specified simplification policy was learned to generate simplified paraphrases that aligned with user preferences.

Sequence editing is a recently explored direction in sentence simplification research. For instance, [13] proposed a sequence-editing framework for sentence simplification, utilizing an edit-action sequence generation model that predicted edit actions, such as deletion, insertion, and substitution, to simplify complex sentences.

2.2.1 ACCESS: AudienCe-CEntric Sentence Simplification

A recent method to approach controlled text generation as sentence simplification involves using control tokens for encoding the transformations between complex and simple text. The ACCESS model by [1] is, at its core, a Transformer model [6]. Like past methods, simplification is treated as a monolingual machine translation task. However, the input (complex) text to the model is first prepended with a set of control tokens, each representing a specific property along which the text is to transform and a floating-point number, which describes the extent to which the transformation is to take place. This modification is shown to improve results beyond just using a Transformer model and achieves SOTA performance.

[1] https://github.com/kvadityasrivatsa/gem_2021_simplification_task

Similar approaches for controlling generated text have been explored in other domains: [14] uses control tokens to estimate and control the amount of hallucination in generated text, [15] explored pre-pending control tokens to the input text for summarization, providing control over the length of the output, and customizing text generation for different sources.

2.3 Control Attributes

In this section, we discuss the control attributes (or control tokens) used by the SOTA method [1] intended for allowing the model to learn relations between the source (complex) and target (simple) texts along the properties encoded by each control token. During inference, these tokens are then set to values that fetch a desired (simplified) paraphrase of the source sentence. Consider the following example:

- **Complex:** `"<NbChars_0.80> <LevSim_0.76> <WordRank_0.79> it is particularly famous for the cultivation of kiwifruit."`
- **Simple:** `"It is mainly famous for the growing of kiwifruit ."`

2.3.1 Text Length Compression

Compression in sequence length has been shown to be correlated with the simplicity and readability of text [16]. Since compression as an operation directly involves deletion, controlling its extent plays a crucial role in the extent of information preservation. We make use of the **compression ratio** (control token: '**NbChars**') between the character lengths of the simple and complex sentences to encode for this attribute.

2.3.2 Paraphrasing

The extent of paraphrasing between complex and simple sentences ranges from a near replica of the source sentence to a dissimilar and possibly simplified one. The measure used for this attribute is **Levenshtein similarity** [17] (control token: '**LevSim**') between the complex and simple sentences.

2.3.3 Lexical Complexity

For a young reader or a second language learner, complex words can substantially decrease the text's overall readability. The average **word rank** (control token: '**WordRank**') of a sequence has been shown to correlate with the lexical complexity of the sentence [18]. Therefore, similar to [1], we use the average of the third-quartile of log ranks of the words in a sentence (except for stop-words and special tokens) to encode for its lexical complexity.

2.3.4 Syntactic Complexity

Complex syntactic structures and multiple nested clauses can decrease the readability of text, especially for people with reading disabilities. To partially account for this, we make use of the maximum **syntactic tree depth** (control token: '**DepTreeDepth**') of the sentence as a measure of its syntactic complexity. We use SpaCy's English dependency parser [19] to extract the depth. The deeper the syntax tree of a sentence, the more likely it is that it involves highly nested clausal structures.

2.4 Experiments

In this section, we describe the experimental setup used to observe the performance of the control tokens described above.

2.4.1 Models

For our initial experiments, we train the following models. Note that all models are architecturally identical and are trained using the same parameters. They solely differ by the data variant used to train them. This is so we can pinpoint the difference in the simplifications generated and the resultant score pertaining to each variant. The models are grouped into the following broad categories:

1. **Baseline:** We train a single Transformer model (labeled T) without any control tokens as a baseline reference. The input data consists of complex sentences, and the corresponding simple sentences are regarded as the target sequences.
2. **Individual Control Tokens:** To gauge the impact of each control token individually, we train four separate Transformer models with one control token each:
   - T + (NbChars / LevSim / WordRank / DepTreeDepth)
3. **Combinations of Control Tokens:** We also investigate the interactions of specific subsets of control tokens using the following three models:
   - T + NbChars + LevSim
   - T + NbChars + LevSim + WordRank
   - T + NbChars + LevSim + WordRank + DepTreeDepth (All)

2.4.2 Datasets

For training, we use the WikiLarge dataset [20], with 296,402 automatically aligned complex-simple sentence pairs obtained from the English Wikipedia and Simple English Wikipedia. For validation and testing, we use ASSET [7] and TurkCorpus [4]. Both datasets contain the same source (complex) sentences in their test (359 sentence pairs) and validation sets (2000 sentence pairs). The more recent dataset i.e. ASSET provides ten human-annotated simplifications for each of the 2,359 source sentences, whereas TurkCorpus provides eight.

Apart from lower-casing all three splits of the data, the data pairs of the trainset with token length lower than three were removed, and sentence pairs with compression ratio ($len(target)/len(source)$) beyond the bounds [0.2, 1.5] were omitted.

2.4.3 Evaluation Metrics

Our model is evaluated on both BLEU [21] and **SARI**. SARI was introduced by [4] as a monolingual metric which "principally compares **S**ystem output **A**gainst **R**eferences and against the **I**nput sentence." The metric is designed to take into account the precision and recall of 3 distinct token-level operations between the source, targets, and prediction texts i.e. add, keep, and delete. For the following equations, I denotes the set of n-grams in the input sequence, O in the output sequence, and R denotes the set of reference sequences provided.

The precision and recall of the **add** operation are given by:

$$p_{add}\left(n\right) = \frac{\sum_{g\in O} \min\left(\#_g\left(O\cap\bar{I}\right), \#_g\left(R\right)\right)}{\sum_{g\in O}\#_g\left(O\cap\bar{I}\right)} \tag{2.1}$$

$$r_{add}\left(n\right) = \frac{\sum_{g\in O} \min\left(\#_g\left(O\cap\bar{I}\right), \#_g\left(R\right)\right)}{\sum_{g\in O}\#_g\left(R\cap\bar{I}\right)} \tag{2.2}$$

where, the binary value $\#_g\left(\cdot\right)$ informs whether the n-gram g has occurred in the input set, and:

$$\begin{gathered}\#_g\left(O\cap\bar{I}\right) = \max\left(\#_g\left(O\right) - \#_g\left(I\right), 0\right)\\ \#_g\left(R\cap\bar{I}\right) = \max\left(\#_g\left(R\right) - \#_g\left(I\right), 0\right)\end{gathered} \tag{2.3}$$

The precision and recall of the **keep** operation are given by:

$$p_{keep}\left(n\right) = \frac{\sum_{g\in I} \min\left(\#_g\left(I\cap O\right), \#_g\left(I\cap\acute{R}\right)\right)}{\sum_{g\in I}\#_g\left(I\cap O\right)} \tag{2.4}$$

$$r_{keep}\left(n\right) = \frac{\sum_{g\in I} \min\left(\#_g\left(I\cap O\right), \#_g\left(I\cap\acute{R}\right)\right)}{\sum_{g\in I}\#_g\left(I\cap\acute{R}\right)} \tag{2.5}$$

where,

$$\begin{gathered}\#_g\left(I\cap O\right) = \min\left(\#_g\left(O\right), \#_g\left(I\right), 0\right)\\ \#_g\left(I\cap\acute{R}\right) = \min\left(\#_g\left(I\right), \#_g\left(R\right)/r\right)\end{gathered} \tag{2.6}$$

and the precision of the **delete** operation is given by:

$$p_{del}\left(n\right) = \frac{\sum_{g \in I} \min\left(\#_g\left(I \cap \bar{O}\right), \#_g\left(I \cap \bar{\acute{R}}\right)\right)}{\sum_{g \in I} \#_g\left(I \cap \bar{O}\right)} \tag{2.7}$$

where,

$$\begin{aligned} \#_g\left(I \cap \bar{O}\right) &= \max\left(\#_g\left(I\right), \#_g\left(O\right), 0\right) \\ \#_g\left(I \cap \bar{\acute{R}}\right) &= \max\left(\#_g\left(I\right), \#_g\left(R\right)/r, 0\right) \end{aligned} \tag{2.8}$$

The metric omits the use of recall on the delete operation as over-deletion is more detrimental to the overall readability of text than not removing tokens at all.

For each operation, the dataset-wide precision, recall, and f1-score are given as follows:

$$\begin{aligned} P_{operation} &= \tfrac{1}{k} \sum_{n=[1,\ldots,k]} p_{operation}\left(n\right) \\ R_{operation} &= \tfrac{1}{k} \sum_{n=[1,\ldots,k]} r_{operation}\left(n\right) \\ F_{operation} &= \tfrac{2 \times P_{operation} \times R_{operation}}{P_{operation} + R_{operation}} \end{aligned} \tag{2.9}$$

$$operation \in [del, keep, add]$$

The final SARI score is given by the arithmetic mean of the f1-scores of all three operations.

$$SARI = \frac{F_{add} + F_{keep} + F_{delete}}{3} \tag{2.10}$$

[1] points out that BLEU favors directly replicating the source sentence because of a high N-Gram similarity between the source and target sentences in most sentence simplification datasets. Therefore we only use SARI to rate and compare the models. We also make use of SARI to choose the best-performing checkpoints on the validation sets of each of the tracks for evaluation on their respective test sets.

2.4.4 Implementation Details

We implement all models using Facebook's FairSequence toolkit [22]. The core model is the Transformer architecture [6] with six encoder and decoder layers, eight attention-heads in each encoder/ decoder module, a 300-dimensional hidden and input vector size, unlike the 512-dimensional model used by [1], and 2048 dimensional fully-connected. We employ the Adam optimizer [23] ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$), with a learning rate of 10^{-4} and 4000 warm-up updates, while dropout is set at 0.2. Each model was trained on 4 Nvidia GeForce GTX 1080 Ti GPUs with 64 GB of vRAM. The training was carried out for 20 epochs and took roughly 11 hours for each model. During inference, we set the control tokens to NbChars: `0.95`, LevSim: `0.75`, and WordRank: `0.75`.

Model	ASSET				TurkCorpus			
	Test		Val		Test		Val	
	BLEU	SARI	BLEU	SARI	BLEU	SARI	BLEU	SARI
T (Baseline)	71.384	33.949	75.947	33.297	74.284	34.529	78.487	34.729
T + NbChars	70.178	34.982	73.197	33.812	73.794	36.299	76.193	36.216
T + LevSim	69.828	35.372	73.476	34.928	73.099	35.040	75.361	35.208
T + WordRank	69.293	35.836	72.996	35.129	73.032	36.823	75.019	36.688
T + DepTreeDepth	70.835	34.751	74.230	34.823	73.902	34.779	77.511	34.930
T + NbChars + LevSim	69.083	36.003	73.004	**36.103**	72.974	37.392	74.329	37.222
T + NbChars + LevSim + WordRank	68.815	**36.707**	72.561	35.992	71.167	**37.801**	74.339	**37.604**
T + All	68.992	36.220	72.846	35.823	71.734	37.018	74.843	36.928

Table 2.1: SARI & BLEU scores on ASSET and TurkCorpus using baseline and control token models.

2.5 Results and Discussion

We report the scores of the baseline model (T) and control token based models on the test and validation sets of the ASSET and TurkCorpus datasets against BLEU and SARI in Table 2.1.

- All single token models (T + NbChars/ LevSim/ WordRank/ DepTreeDepth) score better than the T on SARI. On average, the individual boost in performance by control tokens is given by WordRank > LevSim > NbChars > DepTreeDepth.
- Combining control tokens also improves the overall score in case of T + NbChars + LevSim & T + NbChars + LevSim + WordRank (**$\mathbf{T_{C+Pre+NER}}$** for short). However using all control tokens at once seems to degrade SARI scores across evaluation splits.
- Note however, the scores for BLEU and SARI correlate negatively for all models. This demonstrates how BLEU favors n-gram matching over token level operations, which leads to inflated results for models which produce near replicas of the complex source text.

2.5.1 Quality of Token-to-Attribute Representation

Each control token defined in Section 2.3 is a proxy for a control attribute. Therefore, its is important to observe the resultant relation between the change in the attributes corresponding to their tokens for each model. Figure 2.2 plots the variation in relative extent of each attribute (`attr` in figure) against the range of control tokens (`param` in figure) applied. Note that each row is obtained from the corresponding single-token model.

- The cases of direct attribute-to-token matching are seen in the diagonal cells of Figure 2.2. For attributes Compression Ratio and Levenshtein Similarity, the resultant ratios concentrate around the specified 0.5 and 0.75 values set by the respective control tokens i.e. NbChars and LevSim. However, for Word-Rank ratio and Dependency Tree Depth, this relation does not hold strongly. It is important to note that Word rank is the inverse normalized frequency of the vocabulary, and dependency depth is determined using a separate parsing model, both of which are **not made explicitly available to the mode**l. In comparison, length compression and token level edit distance are easily derivable quantities.

- It was observed that upon setting the length compression attribute to reduce the text length (NbChars <= 0.7), the corresponding models often produced a truncated form of the source text as a trivial output. An example of the same is as follows:

  - **Source:** "`<NbChars_0.6> Wind shear began to dramatically increase and a weakening trend began by early September.`"
  - **Output:** "`Wind shear began to dramatically increase and a weakening trend began.` ~~`by early September.`~~"

  The evidence of this can be seen through the cross-attention scores between the complex and simple texts shown in Figure 2.1. Notice the hightened corresponding attention weights for the NbChars and LevSim tokens near phrasal or clausal endings. This may be indicative of a higher likelihood of truncating the source sentence at such positions. This form of trivial truncation does effectively achieve the desired control token as well as token ratio, but does not improve the readability of the text.

- In many models, in response to the lowering of the word-rank ratio, the outputs tend to **replace named entities with pronouns**. Most named entities occur rarely compared to other content words, therefore have a high word rank. Replacing them with pronouns drastically brings down the average word rank of the text. Though not always an undesirable change, within the limited context of single-sentence simplification, this leads to the dropping of important information. An example of the same is as follows:

  - **Source:** "`<WordRank_0.5> Barack Hussein Obama II is an American former politician...`"
  - **Output:** "`He` ~~`Barack Hussein Obama II`~~ `is an American former politician...`"

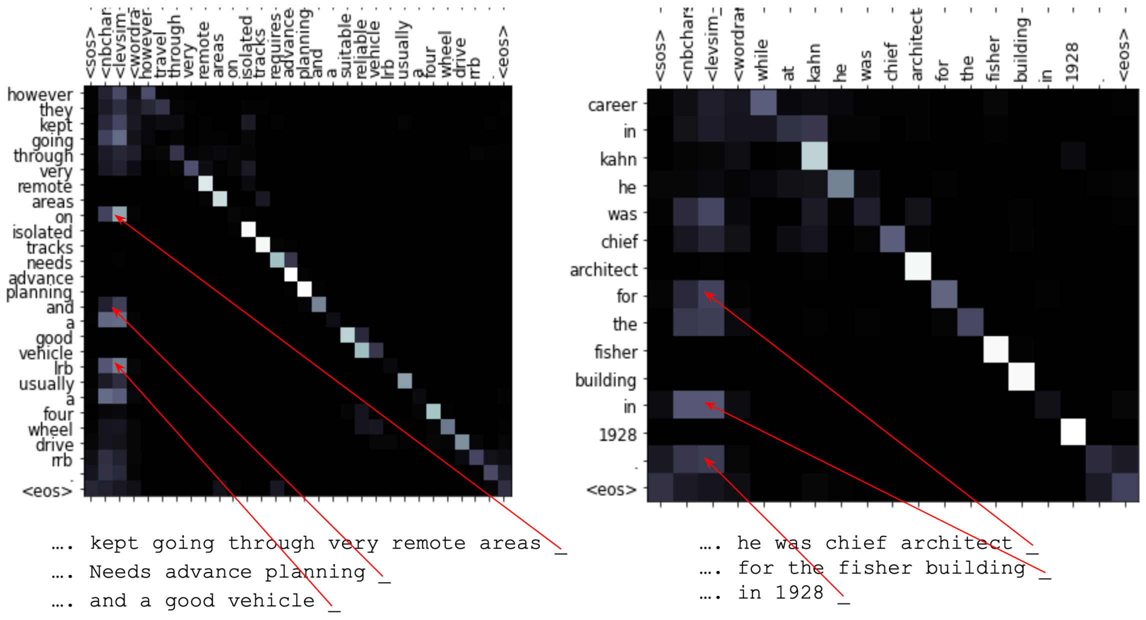


Figure 2.1: Cross-attention plots between complex and simple texts with weights for control-tokens.

2.5.2 Cross Token Influence

In Section 2.5.1, we analyzed the relation between the direct attribute-to-token relation in the model outputs. However, an attribute of the output text can be affected by tokens other than the one designated as its proxy. The non-diagonal cells in Figure 2.2 show this cross-token influence.

- Observe the first row in Figure 2.2 corresponding to the control token NbChars. Upon lowering the target value of NbChars, not does the average compression ratio reduce, but the average levenshtein similarity and dependency tree depth (for NbChars $= 0.5$) also decrease. This is explained by the shorter output sequence length (in some cases due to erroneous truncation), which also increases the levenshtein distance with fewer tokens and reduces the dependency depth with fewer clauses.

- Although the word-rank ratio remains largely unchanged when control tokens other than WordRank are varied, modifying the WordRank token leads to a significant drop in all other attribute ratios.

In an ideal form of token level control, it is prefered that all control tokens can be varied independently with corresponding effects on the text. But as demonstrated above, most control tokens are proxies of attributes which are inter-dependent (compression ratio, levenshtein distance, and dependency depth). This is seen by the high correlation between NbChars and LevSim ($\rho = 0.83$; p-value $< 10^{-3}$) & DepTreeDepth ($\rho = 0.69$; p-value $< 10^{-3}$). This suggests that these tokens cannot be utilized independently, and their individual ranges must be specified carefully to ensure non-trivial outputs.

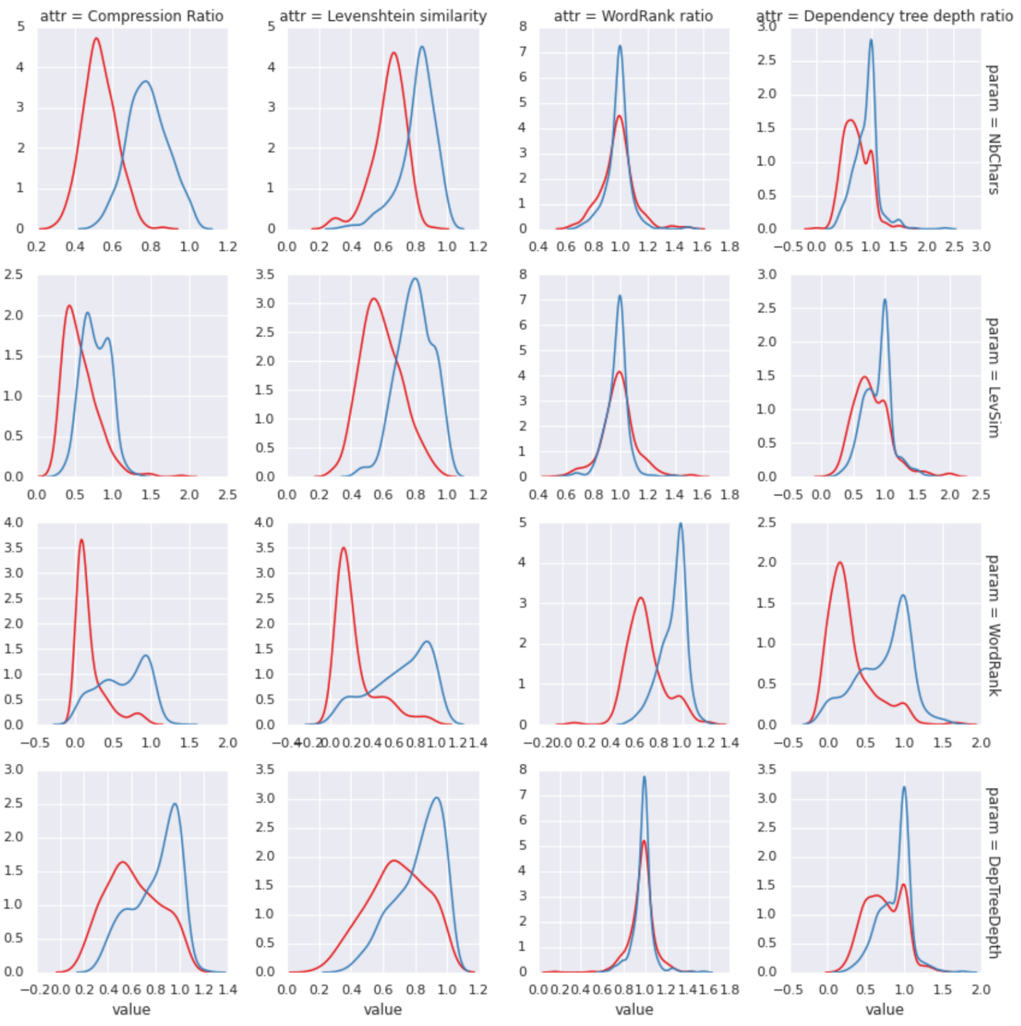


Figure 2.2: Affect of control tokens (params) on control attributes (attr). Plot lines with red and blue colors denote models with control token values set to 0.5 and 0.75 respectively during inference.

2.6 Improving Token Control

In this section, we discuss a few approaches to address some of the pitfalls of the existing models. We apply the following modifications to the best performing model from our initial experiments i.e. T_C (T + NbChars + LevSim + WordRank), followed by performance comparison as well as a qualitative analysis of any potential improvements or flaws induced.

2.6.1 Named-Entity Masking

As shown in Section 2.5.1, models equipped with the WordRank control token often drop the named-entities of the source (complex) text in favor of function words like pronouns to meet the requirement of bringing down the average word rank of the text.

To address this issue, we first make the assumption that for the sake of this task, named-entities in the source text can either be retained in the output text or be dropped entirely, without the possibility of replacement. This allows us to replace individual named-entities with a more generic class of representative tokens to largely isolate them from erroneous replacements.

We thus propose identifying and **masking all named-entities in the source and target text** for the train set, and only the source during inference. During training, we employ an automatic mapping between the named-entities of the source and target text, and provide them corresponding labels as shown in the example below:

- **Source:** `"Sergio P Arez Mendoza ( born January 26 , 1990 in Guadalajara , Jalisco ) , also known as "Checo" P Arez , is a Mexican racing driver ."`
- **NER Masked Source:** `"PERSON@1 ( born DATE@1 in GPE@1 ) , also known as " PERSON@2 " , is a NORP@1 racing driver ."`

We make use of the Ontonotes NER tagger [24] in the Flair toolkit [25]. We identify named entities in the complex halves of all three of the data splits and replace them with one of 18 tags (obtained from the NER tagger). NER replacement for simplification was previously explored by [20], but consists of only 4 classes. The large number of tags allow for a fine division between different named-entity types, permitting more granular contexts while vastly reducing the named-entity vocabulary. Note that while computing the average word-rank of texts, masked tokens are ignored entirely.

The masked data is then used for training and subsequent generation on the test set. Any tags in the simplified output are located in the saved NER-mapping and reverted back to the original token or phrase. This step not only prevents proper nouns from getting replaced, but also greatly reduces the model vocabulary, allowing for faster training.

2.6.2 Pre-Trained Embeddings

Note that all Transformer models discussed uptill now were trained from scratch with randomly initialized embeddings for all tokens in the complex and simple halves. It was also ensured that parameter sharing between the encoder and decoder for the input embeddings is maintained, keeping in mind that the monolingual task at hand can leverage shared vocabulary.

However, the number of sentences and the resultant vocabulary of the WikiLarge dataset is too small to ensure well-representative input token embeddings while training for the task of simplification. Additionally, there may be a large divide between the vocabularies of the complex and simple halves of

Model	**ASSET**				**TurkCorpus**			
	Test		**Val**		**Test**		**Val**	
	BLEU	SARI	BLEU	SARI	BLEU	SARI	BLEU	SARI
T_C	68.815	36.707	72.561	35.992	71.167	37.801	74.339	37.604
T_C + Pretrained	62.488	38.845	71.536	37.700	63.861	38.139	73.627	38.196
T_C + NER	59.215	39.380	70.433	37.985	58.985	38.996	72.181	**38.375**
T_C + Pretrained + NER	59.324	**39.551**	70.202	**38.897**	59.586	**39.777**	68.622	38.231

Table 2.2: SARI & BLEU scores on ASSET and TurkCorpus using the T_C (T + NbChars + LevSim + WordRank) model, its Pretrained and NER variants, and the combined model wherein both Pretrained embeddings and NER masking is applied to T_C.

WikiLarge. Thus there is a need for pre-training the Transformer model or separately providing token embeddings trained on a large corpus of text which contains a mix of complex and simple contexts to allow the resultant embeddings to learn semantic similarities between related complex and simple tokens to ensure better token replacement during simplification.

To address this, we use **FastText's pre-trained 300-dimensional embeddings** [26] as input embeddings for subsequent models. The pre-training provides more-informed embeddings than randomly-initialized vectors for further training, and also extends the unified vocabulary of the models.

2.6.3 Discussion

We apply the above proposed modifications onto the T_C model separately i.e. $T_{C\,+\,NER}$ & $T_{C\,+\,Pre}$ and together i.e. $T_{C\,+\,Pre\,+\,NER}$. We report the score of all modified models along side T_C on the test and validation sets of ASSET and TurkCorpus against SARI and BLEU in Table 2.2.

- All three modified variants outperform T_C on SARI across evaluation sets. Within the modified set, $T_{C\,+\,NER}$ scores superior to $T_{C\,+\,Pre}$, and their combined model i.e. $T_{C\,+\,Pre\,+\,NER}$ performs the best. Again, notice the decreasing value of BLEU scores as SARI increases, indicating outputs with a greater dissimilarity from the source sentence.

- $T_{C\,+\,Pre\,+\,NER}$ shows a **better retention of named-entities** from the source text than T_C, as shown in the first example below:

  - **Source:** `"orton and his wife were happy to have alanna marie orton on july 12, 2008."`
  - **T_C:** `"orton and his wife, dorothy marie orton on july 12, 2007."`

- **$T_{C+Pre+NER}$:** `"orton and his wife supported alanna marie orton on july 12, 2008."`

Both the name `"alanna"` and the year `"2008"` are masked and thus retained in $T_{C+Pre+NER}$, whereas T_C replaces tem with similar named-entities i.e. `"dorothy"` and `"2007"`. Similarly, consider the the second example:

- **Source:** `"aracaju is the capital of the state."`
- **T_C:** `"it is the capital city of the country ."`
- **$T_{C+Pre+NER}$:** `"aracaju is the capital city of the country."`

$T_{C+Pre+NER}$ retains the masked location `"aracaju"`, however, T_C replaces it with the pronoun `"it"`.

- We also identified cases where NER tagging led to errors in the ouput sequence. One such instance is shown below:

  - **Source:** `"yoghurt or yogurt is a milk-based food made by bacterial fermentation of milk."`
  - **$T_{C+Pre+NER}$:** `"yogurt is a type of food that is made by bacterial fermentation of product@1."`

  Upon generating an incorrect named-entity label which cannot be found in the respective NE-mapping for the sample, the label is not replaced with the intended named-entity.

- Similarly, although using pretrained emebddings improved the overall performance of the models, it led to a few cases wherein undesired token replacements take place. As the vector representation of antonyms is often very similar by virtue of a highly similar context in text, certain tokens in the sample are replaced with their respective antonyms to conform to the specified control token. The following is such an instance:

  - **Source:** `"entrance to tsinghua is very very difficult."`
  - **$T_{C+Pre+NER}$:** `"the entrance to tsinghua is very very simple."`

2.7 Conclusion

In this chapter, we investigate the use of control tokens as means for using controlled text generation for sentence simplification. For this, we focus on the model ACCESS [1], which uses control tokens prepended to the complex source sentences to train a Transformer model to learn the transformations between the complex source text and the simple target text along specific attributes. We analyze the claimed relation between each of the control tokens to their respective attributes and found that despite an overall improvement in performance, control tokens do not really help generate simplifications with the desired high-level attributes. We propose two methods to address the pitfalls of the SOTA model, (i) masking named-entities before training and inference to prevent their unwanted replacement as well as to optimize the vocabulary size for the task, and (ii) utilizing pretrained embeddings for the model instead of training from scratch to avoid token sparsity. The proposed methods help improve SOTA scores, better retain named entities, and produce more varied simplifications.

Further research towards curbing the data sparsity induced by the use of separate tokens for the entire range of each control token is required. This may be fixed using a single scalar input to the model for each control token, or to use pretrained or positional embeddings meant to better encode a linear range of values. Using named-entities helped model performance, but also revealed cases where incorrect NER or improper NE mapping leads to erroneous outputs. This can be resolved using a more robust mapping technique instead of the naive method currently used. Additionally, as different datasets (thereby different domains) may involve different styles of simplification, control tokens (or any other token-level or model-level input) to encode for this variation can be incorporated.

Beyond the constraints of the experimentation in this thesis, this task can be perhaps performed with better performance and lower sparsity with the use of newer, pretrained Seq2Seq models such as BART [27].

NOTE: Optimizing sentence simplification methods contributes to our larger goal of improving text accessibility. However, with the advent of recent large language models, having been trained on much larger datasets with a significantly larger parameter count, many tasks such as sentence simplification can now be performed with better control and correctness. However it is important to note that not all NLP tasks equally benefit from these developments, and still require targeted efforts. One such task is **Conversation Disentanglement**, the process of isolating individual conversational threads from a dialogue comprising of multiple active participants at once. This is yet another task which helps make text more accessible for further processing, therefore we explore the work done in this field in the following chapters.

Chapter 3

Conversation Disentanglement

3.1 Introduction

In this chapter, we provide a background into how conversation disentanglement is formally defined by several past works as well as for the scope of this thesis. This is followed by a review of the evolution of the various modeling techniques used to perform disentanglement over time and a detailed description of the existing evaluation metrics utilized for the task.

3.2 Task Formulation

Until now, we have defined conversation disentanglement as extracting distinct threads from a multi-party dialogue. However, it is necessary to formally define the task to better understand the modeling techniques applied or the evaluation metrics used for disentanglement. For the scope of this thesis, we formulate conversation disentanglement in the following manner:

1. Consider a dialogue session consisting of the ordered set of p messages $\mathbb{M}$.

$$\mathbb{M} = \{m_0, m_1, ...m_p\} \tag{3.1}$$

2. All messages are directed between a set of $1 <= q <= p$ unique users (multi-party) $\mathbb{U}$.

$$\mathbb{U} = \{u_0, u_1, ..., u_q\} \tag{3.2}$$

3. Each message is a tuple $m_i = \{s_i, c_i\}$ consisting of the speaker $s_i \in \mathbb{U}$ of the message and the respective text content c_i.

4. Each message m_j must have a parent (or antecedent) message $m_i (0 <= i <= j)$, in response to which the message was posted. Messages with no semantic antecedent, such as dialogue starters or singleton messages, are defined to be their own parents.

5. With this constraint, each dialogue session can be viewed as a graph $\mathbb{G} = (\mathbb{M}, \mathbb{L})$ with messages as nodes and their respective child-to-parent message relations $\mathbb{L}$ as directed edges.

$$\mathbb{L} = \{(m_i, m_j)|0 <= i <= j <= p\} \tag{3.3}$$

6. This allows us to define a conversation or **dialogue thread as a weakly-connected component within the graph** $\mathbb{G}$. We thus define $\mathbb{T} = \{t_0, t_1, ...m_p\}$ as the set of threads within $\mathbb{M}$ as unique and distinct **message partitions**.

7. We thus formalize the task of conversation disentanglement as determining $\mathbb{T}$ from $\mathbb{M}$.

8. The constraint $\mathbb{L} = \{(m_i, m_j)|0 <= i <= j <= p\}$ on message antecedents can become computationally very expensive with large values of p. To tackle this, past methods [3] define a sliding window W to restrict the message-distance between a parent-child message-pair to a maximum of k for each child message m_i.

$$W = \{m_{i-k+1}, ..., mi\} \tag{3.4}$$

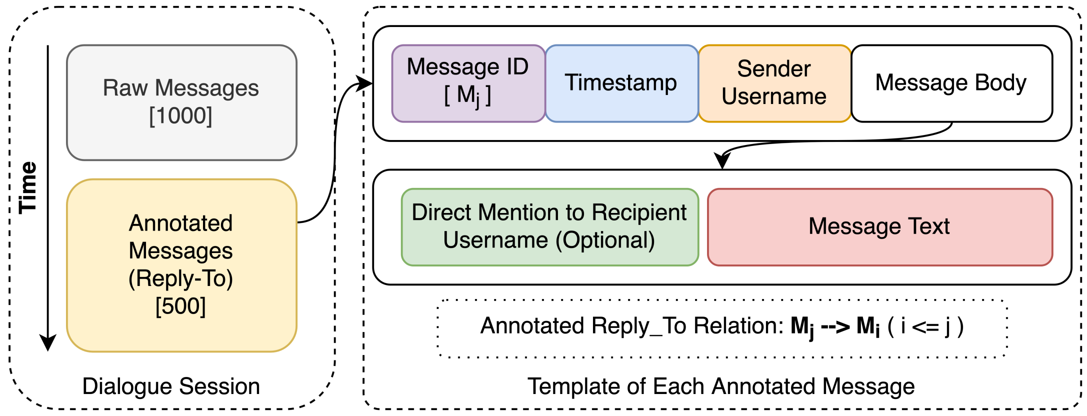
All antecedent relations beyond this window are ignored and replaced with self-loops.

3.3 Datasets

3.3.1 Ubuntu IRC

Research in conversation disentanglement data has primarily focused on data from IRC sources. Significant work in the same started with the use of the `#Linux` IRC channel data [28, 29, 30, 31] by several works for training and evaluation [32, 33, 34]. Following the findings by [35] on the utility of the `#Ubuntu` IRC channel, [36] released a large collection of heuristically extracted 930,000 disentangled conversations sourced from the channel.

[2] showed that only 10.8% of the data extracted by [36] is correct, and subsequently sourced data from `#Ubuntu` (and some from `#Linux`) to release the current largest publicly available & expert-annotated dataset for disentanglement – **Ubuntu-IRC**. It consists of 77,563 messages annotated with *reply-to* relations. The data is structured as 173 (Train: 153 (88.4%); Dev: 10 (5.8%); Test: 10 (5.8%)) individual multi-party dialogue sessions, most containing 500 contiguous annotated messages with an additional context of 1,000 preceding messages to prevent trivial thread initiation at the beginning of each chat session. Inter-annotator conflicts were also resolved with follow-up adjudications.


Time
Raw Messages [1000]
Annotated Messages (Reply-To) [500]
Dialogue Session
Message ID [M_j]
Timestamp
Sender Username
Message Body
Direct Mention to Recipient Username (Optional)
Message Text
Annotated Reply_To Relation: $\mathbf{M_j}$ **-->** $\mathbf{M_i}$ (i <= j)
Template of Each Annotated Message

```
E.g.: #1015 "[02:11] <TheBuntu> L1nuxRules: No response yet."

Annotation: #1015 --> #1012
```

Figure 3.1: Temaplate of Ubuntu-IRC Dataset. On the left, a single dialogue session is shown, with 1000 raw messages for prior context (grey) and 500 subsequent annotated messages (yellow). On the right, the structure of each message is shown: (from left) Message Index, Timestamp, Sender-Username, Message Body (containing an optional direct mention to the recipient username at the beginning).

3.3.2 Other Datasets

Disentanglement data has also been created with data from sources apart from IRC. Some are generated using the existing thread structure present on online platforms like Reddit [37, 38], Slack [39], and Discord [40]. These record message antecedents or thread clusters and later drop this structure. [2] points out that this artificial creation methodology is unlike natural conversations on the respective platforms. Similarly, [41] created the Movie Dialogue dataset by artificially entangling dialogue threads from movie transcripts. Finally, there are smaller datasets (about 7,000 messages) sourced from instant messaging forums (structurally similar to IRC) like Gitter [42], which focus on discussions around Angular, Ethereum, etc.

3.4 Past Modeling Techniques

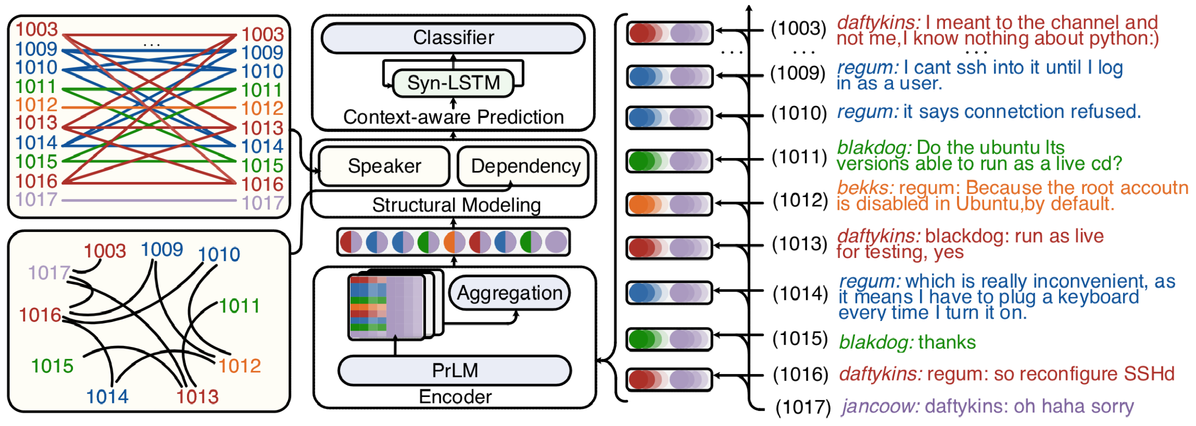
Datasets like Ubuntu-IRC provide explicit & latent information (*non-textual features*) aside from the raw message-body (*textual features*) in a conversation. The expert annotations in the dataset contribute a pivotal explicit feature – *reply-to* relations, which are directed links from each message in a multi-party session to the message(s) it is replying to. Additional non-textual features include message timestamp, author profile (nature & frequency of past messages), *direct mentions* (an explicit reference to the recipient user), running-thread length, and author density (number of unique authors in thread).

1. Various modeling techniques have leveraged these textual & non-textual features in different ways. Initial research uses these features as inputs to statistical classifiers for same-thread classification [28] or for measuring message-level similarity for subsequent clustering [30, 34].

2. Later approaches [2, 43] apply Feed-Forward layers, and Pointer Networks [44] to assist learning higher-order interactions between non-textual and textual features (sampled from pre-trained word-embeddings, e.g., GloVe [45]). However, reported ablation scores show that their performance largely depends on the non-textual features.

3. Recurrent layers like LSTMs [46] have also been used to better encode past message context in early feature-based approaches like [33] as well as end-to-end systems like [41].

4. Finally, the advent of deep contextualized Language Models such as BERT [47] gave rise to two lines of model approaches: Encoding dialogue-level features by fine-tuning LMs [38, 5] or Training LMs on an array of self-supervised dialogue-based tasks and then targeting the shared-knowledge for disentanglement (e.g., MPC-BERT [48]).



(1003) *daftykins:* I meant to the channel and not me,I know nothing about python:)

(1009) *regum:* I cant ssh into it until I log in as a user.

(1010) *regum:* it says connetction refused.

(1011) *blakdog:* Do the ubuntu lts versions able to run as a live cd?

(1012) *bekks:* regum: Because the root accoutn is disabled in Ubuntu,by default.

(1013) *daftykins:* blackdog: run as live for testing, yes

(1014) *regum:* which is really inconvenient, as it means I have to plug a keyboard every time I turn it on.

(1015) *blakdog:* thanks

(1016) *daftykins:* regum: so reconfigure SSHd

(1017) *jancoow:* daftykins: oh haha sorry

Figure 3.2: Architecture of the S4DD (SOTA) model

Recent research has incorporated structural features of conversations in addition to the dialogue-level features: [38] trains a hierarchical Transformer with *reply-to* structure informed masking, DialBERT [5] uses custom conversation-structure & tree-structure based loss functions, and architectures like DAG-LSTMs [49] which are better suited to handle tree-structured data have also been utilized for better encoding structural features [50].

The current SOTA model (S4DD[1]) by [3] on the Ubuntu-IRC dataset is a BERT-based antecedent classifier, which uses an LSTM layer for encoding message context, speaker-specific masked-attention to model speaker-profiles, and a GCN layer [51] to model reference relations like direct mentions.

[1]We name this model S4DD for the ease of reference throughout the thesis

3.5 Existing Evaluation Metrics

Conversation Disentanglement is a partitioning task wherein messages in a dialogue session must be partitioned by their respective threads. Therefore, many of the primary evaluation metrics for disentanglement are measures of distance between two possible clusterings (partitions). The following are the descriptions of some of the metrics used widely in past works.

For all metrics described below, consider X as the true partitioning and Y as the predicted partitioning.

$$X = \{X_1, X_2, ..., X_p\} \tag{3.5}$$

$$Y = \{Y_1, Y_2, ..., Y_q\} \tag{3.6}$$

Note that the total number of elements (messages in our context) remains the same.

$$n = \sum_i |X_i| = \sum_j |Y_j| \tag{3.7}$$

1. **Variation of Information (VI)**: Similar to mutual information but follows the metrics condition of the triangle inequality.

   Let:
   $$p_i = \frac{|X_i|}{n}\ ;\ q_j = \frac{|Y_j|}{n}\ ;\ r_{ij} = \frac{|X_i \bigcap Y_j|}{n} \tag{3.8}$$

   Then the variation of information between X and Y is given by:

   $$VI\left(X;Y\right) = -\sum_{i,j} r_{ij}\left[log\left(\frac{r_{ij}}{p_i}\right) + log\left(\frac{r_{ij}}{q_j}\right)\right] \tag{3.9}$$

2. **Adjusted Rand Index (ARI)**: Calculated along the probability of grouping, therefore uses the number of correctly grouped pairs against all possible ways of pairing.

   Let:

   - a: Number of message pairs that are in the **same** thread in X and **same** thread in Y.
   - b: Number of message pairs that are in **different** thread in X and **different** threads in Y.
   - c: Number of message pairs that are in the **same** thread in X and **different** threads in Y.
   - d: Number of message pairs that are in **different** threads in X and **same** thread in Y.

   Then the Rand Index between X and Y is given by:

   $$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \tag{3.10}$$

To obtain the Adjusted Rand Index, we must normalize the Rand Index (RI) by its expected and max values.

$$ARI = \frac{RI - E(RI)}{max(RI) - E(RI)} \tag{3.11}$$

where,

$$E(RI) = \frac{2\sum_i \frac{1}{2}\binom{p_i}{2}\sum_j \frac{1}{2}\binom{q_j}{2}}{\binom{n}{2}} \tag{3.12}$$

$$max(RI) = \frac{1}{2}\left[\sum_i \frac{1}{2}\binom{p_i}{2} + \sum_j \frac{1}{2}\binom{q_j}{2}\right] \tag{3.13}$$

3. **Adjusted (Normalized) Mutual Information (AMI or NMI)**: AMI differs from mutual information in that it is adjusted for the number of partitions in X and Y, without which, mutual information is always greater for an increasing number of partitions.

$$NMI = \frac{MI - E(MI)}{max(MI) - E(MI)} \tag{3.14}$$

where the mutual information is given by the KL-Divergence between the two groupings.

$$MI\,(X;Y) = D_{KL}\left(P_{(X,Y)} \parallel P_X \otimes P_Y\right) \tag{3.15}$$

4. **Exact Thread Matching (Ex)**: Unlike the above metrics, exact thread matching is not a partitioning metric. It reports the correctness with which clusters in Y match completely with those in X.

$$\#match\,(X;Y) = |Y_j : X_i = X_i\ \exists i \in |X|| \tag{3.16}$$

$$Ex\_P\,(X;Y) = \frac{\#match\,(X;Y)}{|Y|} \tag{3.17}$$

$$Ex\_R\,(X;Y) = \frac{\#match\,(X;Y)}{|X|} \tag{3.18}$$

$$Ex\_F_1 = \frac{2 \cdot Ex\_P \cdot Ex\_R}{Ex\_P + Ex\_R} \tag{3.19}$$

The correctness can be expressed both as Precision (**Ex-P**) and Recall (**Ex-R**), and their combined F1-score (**Ex-F1**).

Chapter 4

Dialogue Properties

4.1 Definition & Motivation

A dialogue (or thread) is a conversation between several participants wherein all individuals are actively involved. Based on the platform or the context in which this dialogue takes place, certain properties of the dialogue and its constituent utterances may vary. These variations are an effect of the specific features or constraints of the platform. For instance, a person having to shout to converse with another individual from afar is constrained along the properties of effort and clarity. The resultant utterances in such a dialogue are typically brief in length (to reduce the effort in communicating) and paced slowly and often segmented at syllable boundaries (to maximize the clarity of the message at the receiver's end). It is also rare for people to speak over each other in such contexts. We define such properties, resultant from the platform-specific constraints and features, as **dialogue properties**. These properties can be organized at the utterance level, such as formality, register, brevity, or at the dialogue level, such as the number of active speakers in a time interval or the average number of distinct conversation threads a speaker is involved in.

By virtue of how dialogues are temporally recorded in a chat forum like IRC, the resultant multi-party conversation (MPC) can be posed as a *noisy room setting*. For each dialogue, interspersed messages & users belonging to other dialogues in the chat room act as external noise that must be eliminated to isolate each thread. As a result, characteristic dialogue properties of the chat room emerge, which can reveal key aspects of the contained multi-party conversations.

4.2 Dialogue Properties of Ubuntu-IRC

In this section, we analyze the dialogue properties of the MPCs within the IRC dataset, which may turn out to be specific artifacts of this medium of conversation, i.e., online chat rooms. Note that in addition to dialogue messages, IRC contains **system-messages**, which include channel-level updates such as a user joining or leaving the channel, changing their display username, etc. We ignore all system messages for subsequent data statistics, experiments, and results as the dataset defines their *reply-to* relations as trivial self-references.

4.2.1 Thread Composition

A single thread (or conversation) is defined as a connected component within the *reply-to* structure of a dialogue session. The thread composition of a dialogue forum can be insightful in understanding its primary use case. The average thread in the train-split of Ubuntu-IRC consists of 6.46 messages (σ=11.70; N=9,273), with 3,553 (38.3%) singleton-threads in the train set (containing only one message) and 61.7% **more-than-one-message threads** that span over 3.5 minutes (σ=70.53; N=5,720) on average and across 55.6 messages (σ=111.55; N=5,720) in the dialogue session.

A large number of **singleton threads** is probably explained by the `#Ubuntu` & `#Linux` channels being primarily help channels, where new users often post queries that do not receive a reply. Additionally, the average thread contains 2.02 users (σ=1.45; N=9,273), but about 4,610 (49%) threads only contain one user.

4.2.2 Message Length

The average (space-separated) word length of messages in IRC is 10.24 messages (σ=9.63; N=59,347), with over **30% of messages containing less than five words**. About 8% of messages are **single-word comments**, often used to initiate or close threads or user-pair interactions, e.g., `"hello"`, `"thanks"`, `":)"`, indicating that emoticons are also used as meaningful units of conversation in IRC.

4.2.3 User Engagement

As mentioned before, the dataset contains messages from help channels, which cater to two broad categories of users: One that primarily asks queries and the other that addresses them. These categories can be roughly observed by the following distributions. Each session in IRC contains 50.43 unique users (σ=20.77; N=153) on average within the annotated span of messages.

The **average number of threads a user participates in**, within a single session is 2.42 (σ=2.56; N=7,716). However, over 51% (3,924) of this **users participate in only one thread** (likely users who only post queries), with just 8.8% (686) of the **users involved in more than five threads** in a session (likely users who regularly answer queries or are channel moderators).

4.2.4 Thread & User Density

In a multi-party chat room like IRC, users can initiate and participate in their respective thread(s) independently of other running threads. This leads to a non-uniform thread density i.e., **density of concurrent threads**. The train-split contains 17.05 (σ=8.84; N=59,374) parallel active threads on average in a dialogue window (W) containing 50 contiguous messages ($|W| = 50$) and about 7.18 non-singleton threads (σ=2.60; N=59,374). A non-uniform density of user participation is also observed along with thread density. The distribution of the **number of active users in a dialogue window** ($|W| = 50$) is shown in Figure 4.1, with a mean of 14.32 users (σ=3.67, N=59,374).

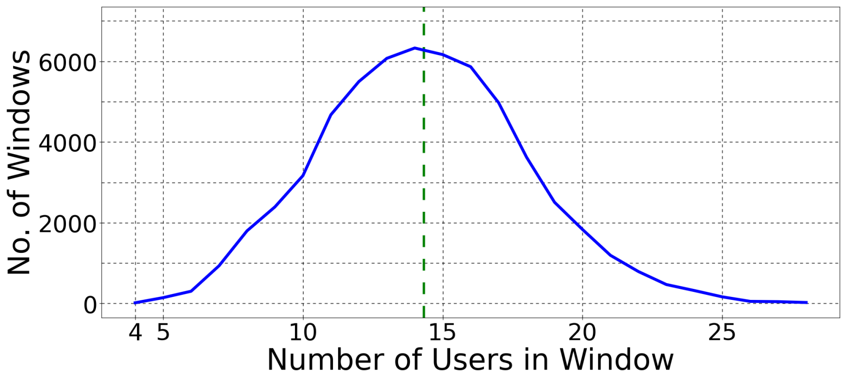


Figure 4.1: User Density i.e. number of active users in a dialogue window of length 50, in Ubuntu-IRC

4.2.5 User Activity

User activity for a given dialogue window W is defined by the **number of constituent messages in the window by the child-user** (i.e., the author of the last message in the window). The average user activity in IRC in a window of 50 messages ($|W| = 50$) is 5.64 (σ=4.77; N=59,347). Figure 4.2 shows the proportion of dialogue windows across user activity.

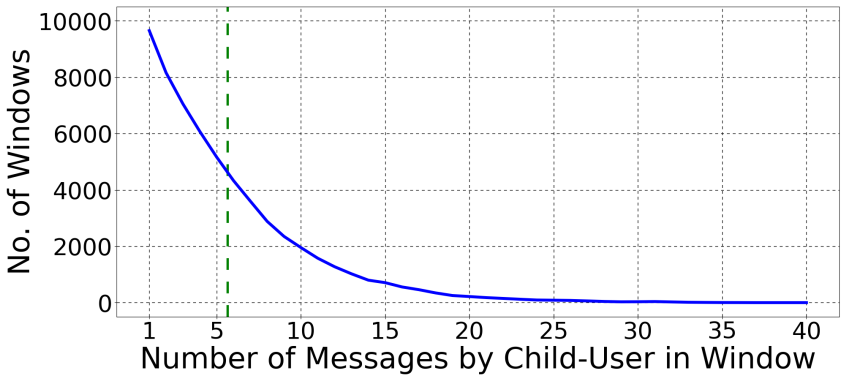


Figure 4.2: User activity i.e. number of constituent messages in a window by the child-user (as the speaker), in Ubuntu-IRC

4.2.6 Antecedent Distance

Antecedent distance (d) refers to the **number of messages between a message (child) and its *reply-to* parent**. The mean antecedent distance is 7.34 messages (σ=31.79; N=59,347), as shown in Figure 4.3 (marked by a green-dotted line). The data is dominated by self-referential (d=0: 24.8%) and previous-message reference (d=1: 18.1%) cases, with 87% of parent messages lying at most ten messages away from their respective children. This can lead to the inflated performance of trivial policy models, which always predict the previous message as the antecedent, as shown in Table 4.1. Considering this distribution, many past works restrict the antecedent-prediction task to a small window of recent messages.

Model	**VI**	**ARI**	**Ex-P**	**Ex-R**	**Ex-F1**	**1-1**	**NMI**
Gold (|W|=50)	98.30	89.06	81.59	89.61	85.41	93.08	97.19
Agreement [2]	95.00	–	–	–	49.50	83.80	–
Previous [2]	66.20	–	–	–	0.00	23.70	–

Table 4.1: Scores obtained on Ubuntu-IRC [2] using gold labels (with a sliding window of last 50 messages), reported inter-annotator agreement, and trivial previous-message as parent policy.

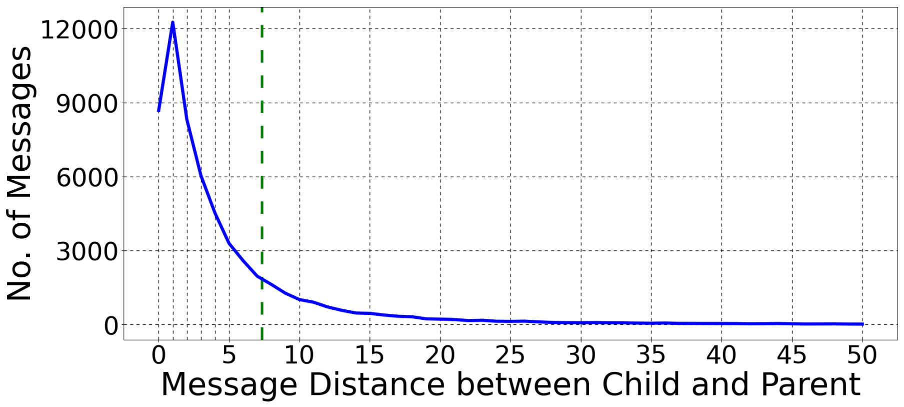


Figure 4.3: Antecedent distance i.e. number of messages between a message (child) and its *reply-to* parent, in Ubuntu-IRC

4.2.7 User Referencing

User referencing (or pointing) is the act of mentioning one or more of the users within the dialogue session in a message. IRC contains an explicit form of referencing, i.e., **direct mentions**, which is used to specify a message's recipient at the start of the message body as shown in Figure 1.1. Though an informal practice, over 44% of IRC samples contain direct mentions. Another kind of user referencing is **soft-mentioning**. Unlike direct mentions, these can occur within the message body as part of the natural text. They don't always point to the message's recipient and may not point to users within the running thread. Users include simplified or abbreviated versions of usernames within the text, making soft mentions harder to detect using simple rule-based methods. The dataset contains 4,166 soft mentions (exact username matches), of which 2,724 mentions (65.38%) point to the recipient user.

4.3 Role of Direct Mentions in IRC

Direct mentions occur in almost half the messages in the IRC data despite being an informal practice. They inform of message recipients, making it easier to track user interactions. Therefore, it is important to investigate where and why users choose to use (or omit) this feature while conversing. Thus, in the following analysis, we attempt to understand the role of direct mentions in the IRC data along with some key dialogue properties mentioned above.

4.3.1 Along Antecedent Distance

The likelihood of using direct mentions in messages from IRC across a range of antecedent distances is reported in Figure 4.4. It shows a sharp increase in direct mention usage as the antecedent distance increases from 0, with about 65% of child messages containing a direct mention and almost 80% of either of the messages in a parent-child pair being directed.

As the distance between the parent and the child message increases in a dialogue session, it can become more difficult for a user to ascertain their *reply-to* relation due to the increasing number of unrelated messages (and thereby, users and threads) occurring between them.

A higher number of direct mentions as antecedent distance increases might play a role in alleviating this difficulty. Thus we expect a monotonic rise in the usage of direct mentions with increasing antecedent distance. However, the likelihood decreases for longer distances.

We find that the average number of users in each thread decreases as the antecedent distance increases. This reduces the need for explicit user-referencing and may also explain why the proportion of direct mentions is lower for longer distances.

This explanation is supported by the strong and significant correlation (Spearman $\rho = 0.59$; p-value < 0.001) between the average number of active users in a thread and the direct mention likelihood for longer antecedent distances (> 5 messages).

4.3.2 Along User-Density & User-Activity

We consider the number of users in a dialogue window ($|W| = 50$) as a proxy for user density in localized time-frames in a dialogue session. Figure 4.5 shows the percentage of directed messages in a

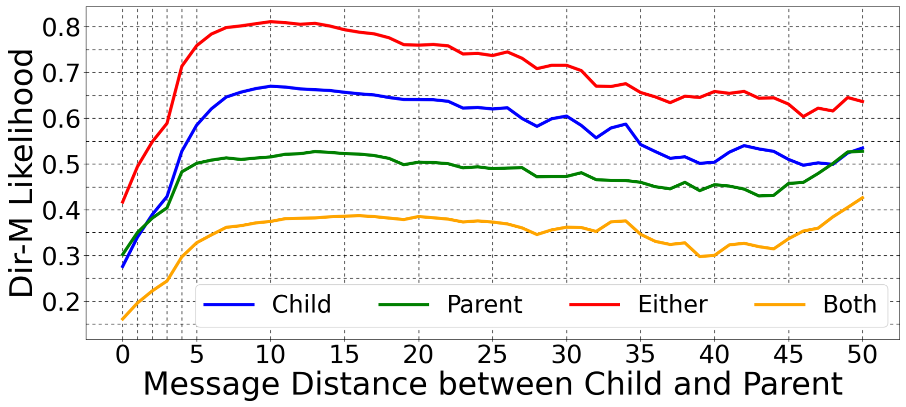


Figure 4.4: Direct mentions along antecedent distance

window ($|W| = 50$) across an increasing number of unique users. We see that users add direct mentions more often as the number of users increases in the message vicinity.

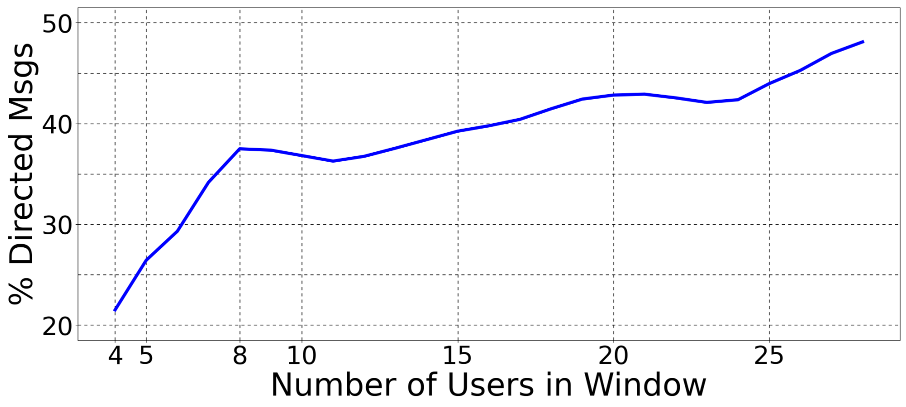


Figure 4.5: Direct mentions along dialogue density

Correspondingly, Figure 4.6 shows the percentage of directed messages in the window in such cases. As the number of messages by the child user dominates a window, the average concurrent threads and unique users in the window predictably decrease, reducing the need for explicit user-referencing.

Ubuntu-IRC is currently the only large-scale expert-annotated dataset for conversation disentanglement, making it an important resource for training and evaluating methods within the IRC domain and for domains such as meeting transcripts & movie dialogues that lack a similar large-scale high-quality dataset.

This chapter shows that direct mentions occur abundantly in the IRC dataset and play a key role along several dialogue properties. However, direct mentioning is primarily a forum-specific (though informal) artifact of Internet Relay Chat and similar online chat forums.

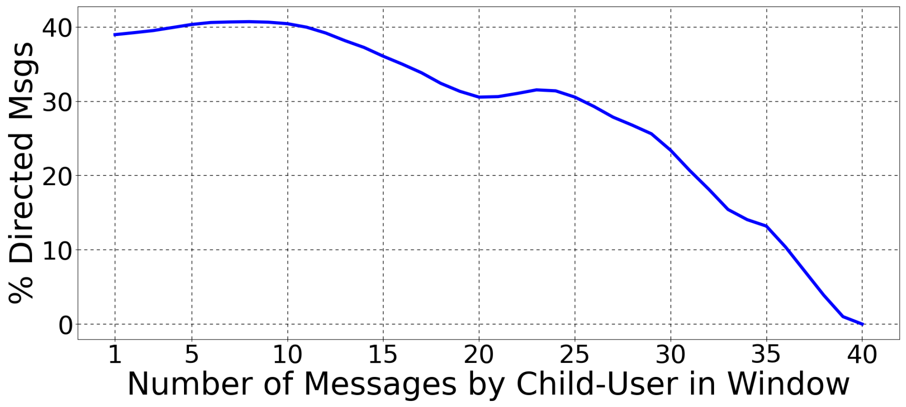


Figure 4.6: Direct mentions along user activity

We suspect this increases the specificity of the Ubuntu-IRC dataset, impeding its current role as the generalized resource for research in disentanglement. To further investigate this specificity, we aim to measure direct mentions' impact on the overall performance of the models trained using Ubuntu-IRC.

Chapter 5

Ubuntu-IRC Hard

5.1 Modifying Ubuntu-IRC

To compare the model performance on the IRC dataset against the absence of direct mentions, we create a modified variant of IRC without direct mentions. Non-system messages in IRC consist of the posting timestamp, the sender's username (at the time), and the message body, as shown in the template below:

```
"[timestamp] <username> message-body"
```

We define a direct mention in the message-body as a reference to a past user (using their exact latest username), strictly at the beginning (first space-separated segment) of the message-body. Following are the steps to reproduce our data with direct mentions removed:

1. Remove any special characters padding the first space-separated token in the message body if they belong to the fixed set `[.,;:?!]`.

2. Compare this isolated token against the usernames of the users that have posted any messages in the corresponding session *before* the current message.

3. It is an instance of direct mention if an active user is found referenced. Accordingly, remove the direct mention i.e. all characters of the first token from the message body (including any padded special characters & white space).

We found a total of 29,190 direct mentions (44.03%) in Ubuntu-IRC (Train: 26,349 (44.39%); Dev: 931 (40.0%); Test: 1,910 (41.35%)). We have also described the estimated distribution of soft mentions in IRC in Section 4.2.7. However, experimentation without soft mentions would require further investigation (beyond the scope of the current work) into possible placeholder strategies as simply deleting them can hamper the flow of natural text. We release this modified variant of IRC as **Ubuntu-IRC-Hard** (or IRC-Hard) publicly.

5.2 Experiments

Conversation Disentanglement has often been formulated as a partitioning/clustering task, i.e. partitioning a given session into mutually exclusive threads. With message-level *reply-to* labels available for IRC & IRC-Hard, we follow past works in segmenting the session into smaller dialogue windows of contiguous messages ($|W| = 50$), wherein the parent(s) of the child message (i.e., last message of the window) must be predicted. As there are only a few cases in Ubuntu-IRC with multiple antecedents of a message ($< 1\%$) [2], we limit the task to predicting a single parent (closest to the child message).

5.2.1 Baseline Models

We devise a set of baselines to better compare the contributions of modeling approaches sourced from past works:

(1) BERT(`[CLS]`) or BC: This model utilizes the `[CLS]` token representation of the message-pairs (separated by `[SEP]` token) obtained from BERT (`bert-base-uncased`) [47] similar to [5]. This is then passed through a linear classification layer (with output dimensionality equal to the prediction window size ($|W| = 50$)) to predict the optimal antecedent.

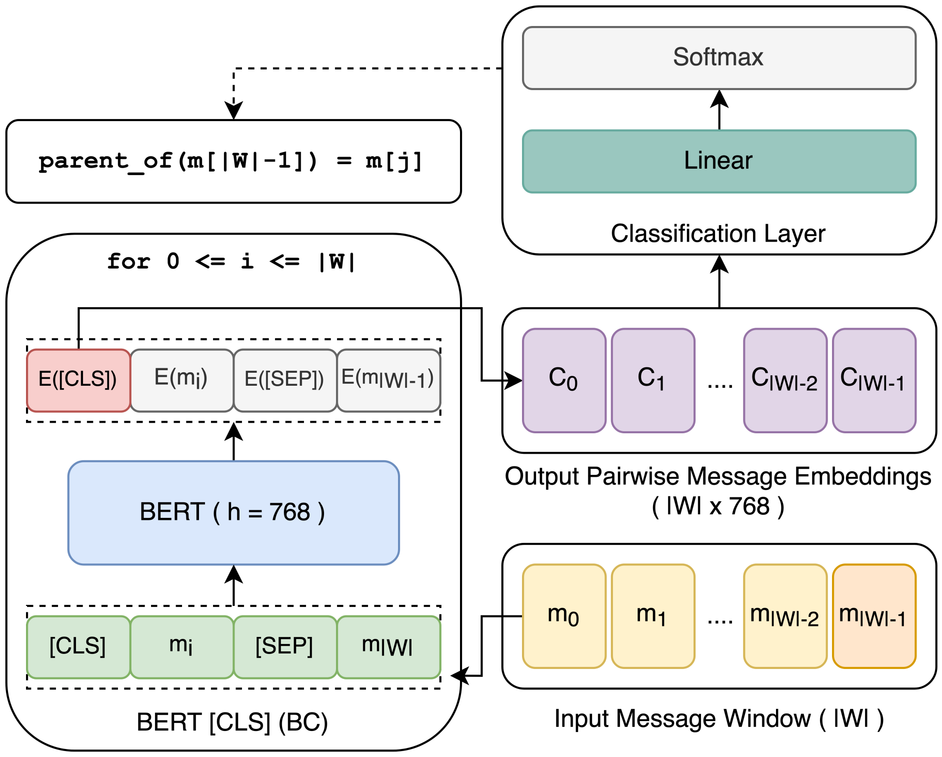


Figure 5.1: Architecture of the BERT-CLS model

+LSTM: Similar to a few past methods [33, 41], this model uses an LSTM for better context awareness. The `[CLS]` token representation from BC is first passed through a single-layered LSTM before applying the classification layer.

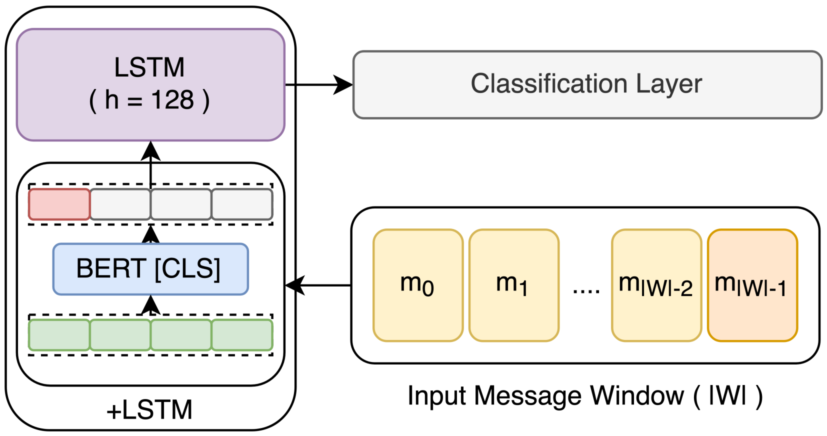


Figure 5.2: Architecture of the +LSTM model

+SMA: For better speaker-profile modeling, [3] use a speaker-masked-attention layer, allowing each message to pay attention to only the messages by the same author (within the prediction window). This model applies the attention layer in +LSTM just before the LSTM layer. In the +SMA pipeline shown in Figure 5.3, messages m_0 & $m_{|W|-1}$ are by speaker U_A and m_1 & $m_{|W|-2}$ by U_B. In such a case, messages m_0 & $m_{|W|-1}$ cannot attend to each other but can attend to others.

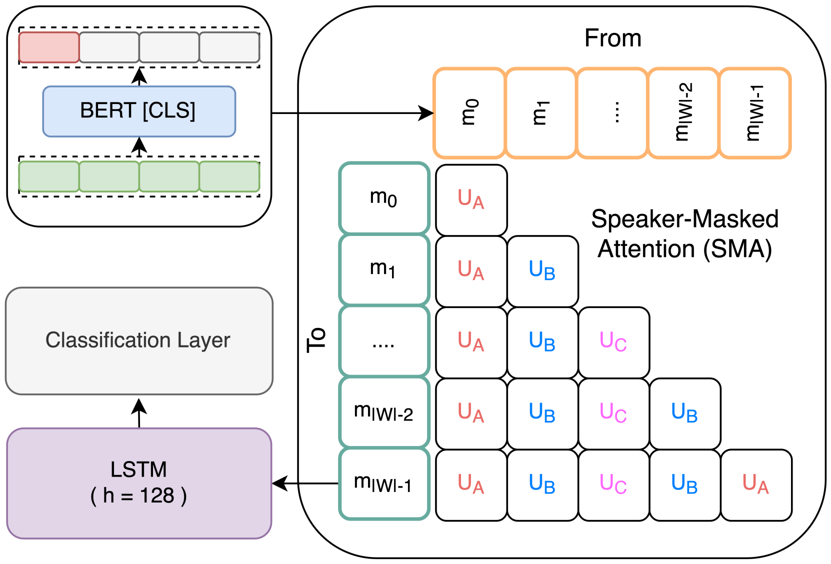


Figure 5.3: Architecture of the +SMA model

+GCN: To better encode reference dependency (direct & soft mentions) [3] use a GCN layer that accepts a graph of reference links between the messages in the prediction window. We apply this layer parallel to the attention layer in +SMA and concatenate their outputs before passing them through the subsequent LSTM layer.

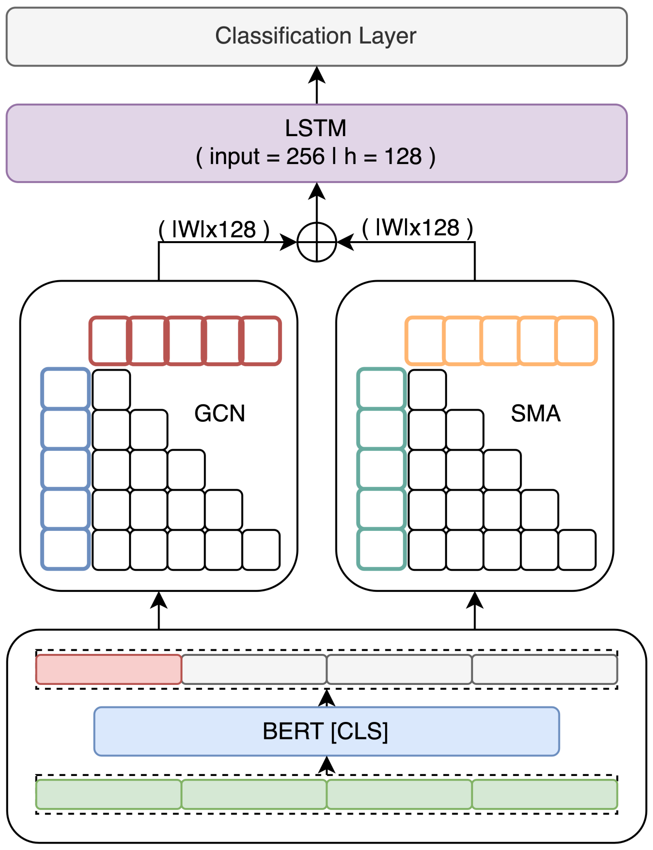


Figure 5.4: Architecture of the +GCN model

BERT(Mean) or BM: Aside from baselines with specific modeling layers, this model uses the mean-pooled output of the token-level representations from BERT instead of the `[CLS]` token representation as in BC.

The output predictions from the classification layer are compared against the gold annotations using a Cross-Entropy loss for all baseline models.

5.2.2 Evaluation Metrics

After antecedent-prediction for each message in a session, the connected components of the *reply-to* relations between messages are evaluated as predicted threads against the gold-annotated threads. We use the following partitioning metrics: Variation of Information (**VI**) [52] ($1 - VI$ for higher scores to indicate better alignment), Adjusted Rand Index (**ARI**) [53], One-to-One overlap (**1-1**) [28], and

Normalized Mutual Information (**NMI**). We also use Precision (**Ex-P**), Recall (**Ex-R**), and F1-score (**F1**) on the exact matching of threads.

We also conduct a qualitative evaluation of the models along several dialogue properties. For this, we define a new metric – **Thread Correctness (TC)**. Note that many of the dialogue properties mentioned in the previous section are defined around the sliding-window formulation of antecedent classification instead of the thread-centric view of conversations. This shift to a local context (window) requires a metric which can be applied to such a context, where existing thread partitioning metrics cannot be applied. Thread correctness measures whether the messages within a dialogue window are correctly grouped with the child-message in the same thread. The metric reports the F1 score on this classification for a given window size. Below is the formulation for TC:

$$P = \frac{|m : T(m)_{pred} = T(m)_{true} = T(m_C)_{true}, m \in W|}{|m : T(m)_{pred} = T(m_C)_{true}, m \in W|} \tag{5.1}$$

$$R = \frac{|m : T(m)_{pred} = T(m)_{true} = T(m_C)_{true}, m \in W|}{|m : T(m)_{true} = T(m_C)_{true}, m \in W|} \tag{5.2}$$

$$Thread\,Correctness = \frac{2PR}{P+R} \tag{5.3}$$

where m is a message within the prediction window W. m_C is the child (last) message in W. $T(m)_{true}$ & $T(m)_{pred}$ respectively denote the actual and predicted thread to which m belongs. P (precision) & R (recall) are combined to generate the F1-Score of TC.

5.2.3 Implementation Details

We train all past models on four initialization seeds (with their average scores reported), adhering to their respective instructions for reproducibility. Baseline and experimental models were trained using an AdamW optimizer [54]. Loss on the dev set was used to stop baseline model training with a patience of 2 epochs. All experiments were performed using 2 Nvidia 1080Ti GPUs with 20GBs of memory. Each training session for S4DD, baseline, & experimental models took about 26-29 hours to complete. Input sequences were truncated (or padded) to 128 tokens, and the dialogue window W for antecedent prediction was set to the last 50 messages ($|W| = 50$). Our baseline and experimental models were implemented using PyTorch [55] and the Transformers library [56].

5.3 Results and Discussion

We report the scores of past and baseline models on both Ubuntu-IRC (IRC) and Ubuntu-IRC-Hard (IRC-Hard) against partitioning metrics in Tables 5.1 and 5.2. A significant drop in performance for all models is recorded on IRC-Hard compared to IRC across metrics. Figure 1.2 shows how S4DD's performance degrades gradually as the proportion of **direct mentions** decreases.

- The feature-based feed-forward model (FF) by [2], shows a steep reduction (by 64.8%) in the exact thread matching (Ex-F1). We observe that FF relies on the number of direct mentions between message pairs as a feature. Removing direct mentions predictably makes this feature unusable, partly explaining the drop. As the GCN layer in S4DD encodes soft mentions in addition to direct mentions, it is slightly more resilient to dropping direct mentions in IRC-Hard (52.8% drop in Ex-F1).
- The baseline models BC & BM also show a similar reduction in scores, although BM drops further and scores worse than BC across VI, ARI, and 1-1. +LSTM, +SMA, and +GCN report increasingly higher performance than BC for IRC. However, this gain is notably diminished upon removing direct mentions (in IRC-Hard).
- The overall drop in VI for most models is much lower in comparison to other metrics. This is explained by a large number of previous-message antecedents (20.65%) in IRC (and IRC-Hard). Most models learn this distribution, resulting in an inflated VI e.g. a trivial system with previous messages predicted as *reply-to* parents fetch a VI of 66.20.

5.3.1 Evaluation Along Dialogue Properties

Current partitioning metrics report model performance aggregated across all threads in the evaluation set. This hides internal performance variations across different dialogue properties. To highlight these variations, we report the TC (see Section 5.2.2) of S4DD and the baseline models across the ranges of several dialogue properties in Figures 5.5, 5.6, 5.7, and 5.8 (wherein, solid lines and bars correspond to models trained on IRC, and dotted lines and slashed bars denote the models trained on IRC-Hard).

- Figure 5.5 shows how the general model performance decreases as the **antecedent distance** increases, where models trained on IRC-Hard perform consistently worse than on IRC for the distance range 0-30. However, S4DD and +GCN trained on IRC-Hard outperform some IRC models beyond a distance of 30 messages. This is probably explained by a much lower proportion of direct mentions in these ranges (for IRC-trained models to utilize), where the additional GCN layer in S4DD & +GCN still allows leveraging soft mentions in lieu of direct mentions.
- A higher **user density** in a dialogue window warrants better user-referencing, leading to a higher proportion of direct mentions, as shown in Figure 4.5. Figure 5.6 shows that models trained on IRC-Hard perform significantly worse as user count increases. Models trained on IRC, in comparison, perform more consistently across user counts.
- A higher **user activity** tends to occur in windows with fewer concurrent threads, therefore, requires lesser user-referencing. Though models score similarly in windows saturated by child-messages on both IRC & IRC-Hard, Figure 5.7 shows that models perform significantly worse on IRC-Hard as the message count decreases.

Group	Model	VI	ARI	Ex-P	Ex-R	Ex-F1	1-1	NMI
Past	FF [2]	92.65	71.91	38.16	40.85	39.46	78.44	87.96
	S4DD [3] (SOTA)	93.47	69.90	45.59	47.57	46.56	80.66	88.85
Baseline	BC	91.52	62.88	32.98	39.64	36.00	75.14	84.98
	BM	91.59	60.79	35.50	41.94	38.45	74.64	85.29
	+LSTM	91.24	63.36	30.98	37.08	33.76	74.40	84.56
	+SMA	92.74	67.08	41.13	44.50	42.75	77.70	87.54
	+GCN	93.21	69.96	40.64	45.52	42.94	79.64	88.20
Weighted Loss	$p = 0.2$	93.68	69.87	**49.25**	**50.38**	**49.81**	80.16	89.28
	$p = 0.4$	93.54	70.08	44.52	47.83	46.12	80.50	88.88
	$p = 0.6$	92.88	73.05	39.09	39.39	39.24	80.30	87.90
	$p = 0.8$	**93.84**	**75.93**	43.06	46.04	44.50	**82.38**	**89.40**
Ensembling	Voting (All Models)	93.25	69.43	45.39	49.10	47.17	79.80	88.41
	IE(U-Count)	93.51	70.85	47.51	48.85	48.17	80.14	88.97
	IE(M-Count)	93.60	70.93	46.78	48.34	47.55	80.68	89.10
	IE(SM)	93.69	72.77	47.25	48.34	47.79	80.80	89.29

Table 5.1: Experimental results on Ubuntu-IRC [2]. Results in bold are the best-observed scores under respective metrics for each dataset variant.

Group	Model	VI	ARI	Ex-P	Ex-R	Ex-F1	1-1	NMI
Past	FF [2]	83.57	45.65	13.66	14.08	13.87	59.34	73.20
	S4DD [3] (SOTA)	86.15	53.90	21.03	23.02	21.98	65.70	76.42
Baseline	BC	85.14	45.17	15.47	18.16	16.71	61.42	73.74
	BM	83.73	44.75	13.16	16.62	14.69	59.64	70.72
	+LSTM	83.97	42.25	12.09	16.11	13.82	60.18	70.83
	+SMA	85.47	53.83	17.85	17.39	17.62	64.28	75.69
	+GCN	86.10	53.75	16.21	18.16	17.13	66.06	75.93
Weighted Loss	$p = 0.2$	86.87	55.56	19.95	22.25	21.04	67.42	77.34
	$p = 0.4$	85.94	54.33	21.57	21.74	21.66	66.42	76.26
	$p = 0.6$	86.34	53.71	21.11	21.48	21.29	66.68	76.79
	$p = 0.8$	85.87	52.64	18.55	20.97	19.69	64.80	75.63
Ensembling	Voting (All Models)	86.81	55.10	22.70	24.55	23.59	66.30	77.30
	IE(U-Count)	86.77	56.01	21.69	23.02	22.33	66.60	77.46
	IE(M-Count)	**87.33**	**56.87**	**24.15**	**25.32**	**24.72**	**68.16**	**78.42**
	IE(SM)	86.97	56.58	22.47	22.76	22.62	66.90	77.93

Table 5.2: Experimental results on Ubuntu-IRC-Hard (See Chapter 5). Results in bold are the best-observed scores under respective metrics for each dataset variant.

- We also compare model performance on samples by **whether the messages contain a soft-mention** in Figure 5.8. Without direct mentions, TC worsens significantly more in the absence of soft mentions. Additionally, worse scores in cases without soft mentions for models other than S4DD and +GCN (which contain dedicated reference modeling layers) show that BC, BM, & +LSTM also leverage soft mentions (although to a smaller extent).

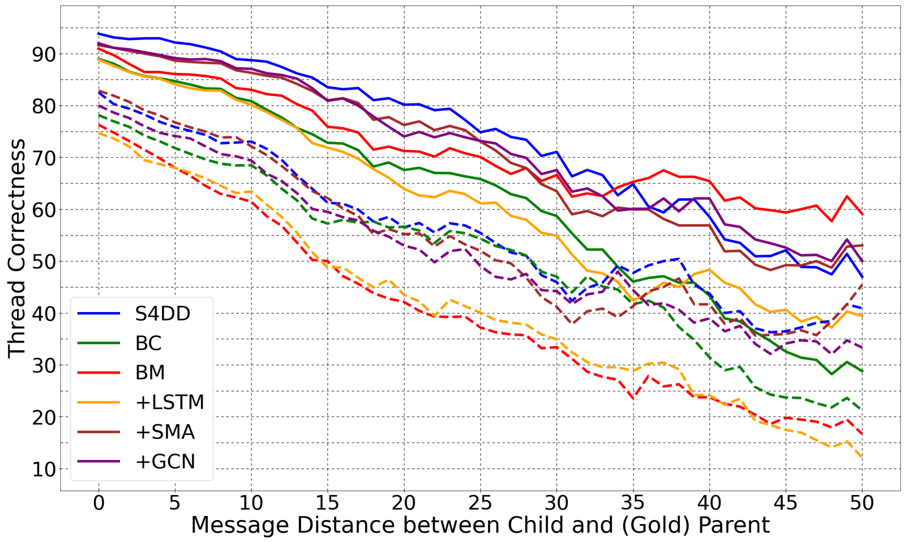


Figure 5.5: TC of past and baseline models along antecedent distance

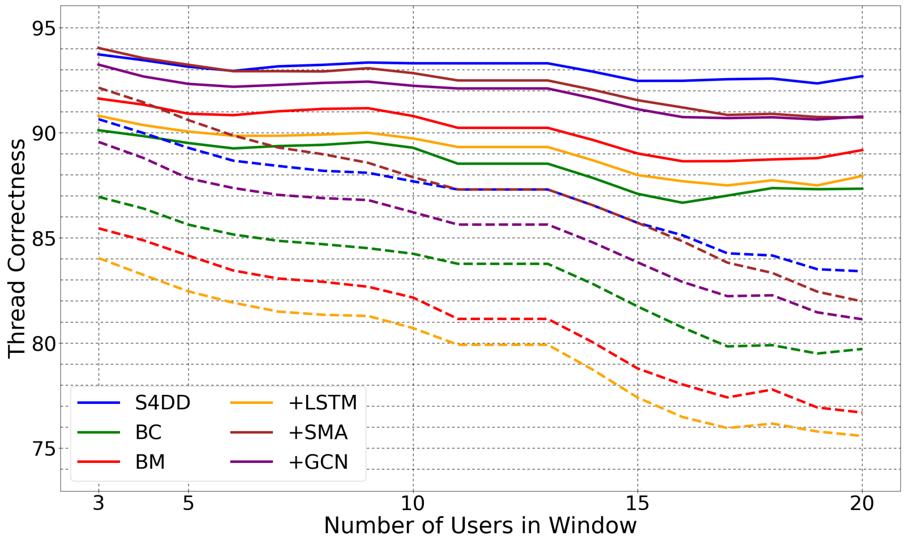


Figure 5.6: TC of past and baseline models along user density

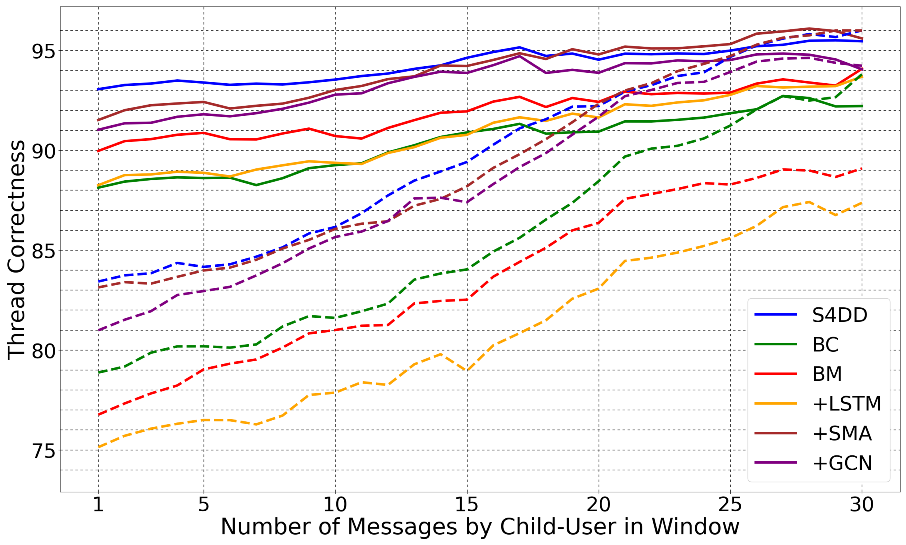


Figure 5.7: TC of past and baseline models along user activity

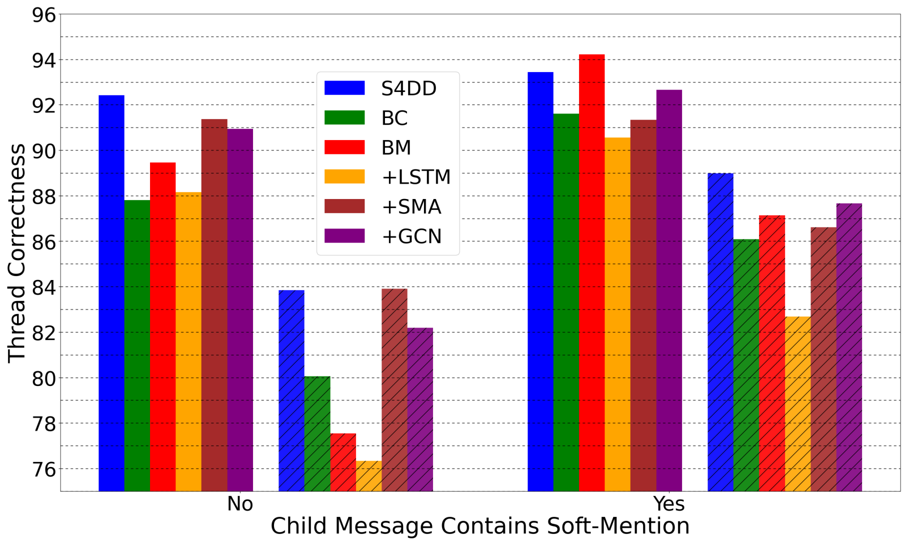


Figure 5.8: TC of past and baseline models along soft mentioning

Chapter 6

Dialogue Property Aware Modeling

6.1 Motivation

A key takeaway from the investigations in section 5.3.1 is that models can have varied capabilities for different sub-ranges of dialogue properties. This chapter aims to leverage these variations for existing SOTA methods to improve their performance and overall generalizability.

6.2 Property-Specific Modifications

Qualitative metrics can help identify specific property sub-ranges where generally good models perform poorly. This localized brittleness is partly due to a non-uniform representation of different property sub-ranges in the data, e.g., longer antecedent distances are rare in IRC, and corresponding models report a significant performance drop as distance increases (although not solely due to fewer representative data instances).

We thus propose a **weighted loss formulation** to inversely scale the training loss for each data instance according to the representative proportion of its context in the data along a dialogue property. We experiment with this approach in the case of antecedent distance. The corresponding loss factor is defined as follows:

$$\beta_d = \frac{\sum_{k=0}^{|W|} |\{m : D_{ant}(m) = k; m \in M_{train}\}|}{|\{m : D_{ant}(m) = d; m \in M_{train}\}|^p} \tag{6.1}$$

where m is a message from the train-split (M_{train}), $D_{ant}(m)$ denotes the message distance between m and its antecedent message, and W is the prediction window with a fixed size (taken as 50). The loss pertaining to each message in the prediction window is scaled by β_d based on its distance (d) from the child (last) message in W. However, note that a highly underrepresented distance will accordingly receive an extremely large weight. This may negatively impact training, with a few individual examples having over-amplified losses. To alleviate this issue, the loss factor is softened using a variable power factor p ($0 < p < 1$), with lower values resulting in a relatively softer scaling factor than for $p = 1$.

6.2.1 Discussion

We train four instances of S4DD on each dataset (IRC & IRC-Hard) using the weighted loss with the following values of p – $\{0.2, 0.4, 0.6, 0.8\}$ and report scores against partitioning metrics in Table 5.2. For a qualitative perspective, Figure 6.1 reports TC along antecedent distance for all models alongside another S4DD instance (without weighted loss) for each dataset.

- Among IRC models, $p = 0.2$ has the highest Ex-P, R, &F1, and $p = 0.8$ scores the highest across all other metrics. Observing their TC along antecedent distance reveals that $p = 0.2$ (`W_0.2`) scores similarly to S4DD for shorter distances and surpasses S4DD for distances roughly > 35 messages.
- A sharper scaling using $p = 0.8$ (`W_0.8`) surpasses S4DD performance after relatively shorter distances i.e., 20 messages, although the scores decrease again for distances > 40.
- Among IRC-Hard models, $p = 0.4$ (`W_0.4`) and $p = 0.6$ (`W_0.6`) have the best Ex-F1 and consistently score better than S4DD across all antecedent distances.

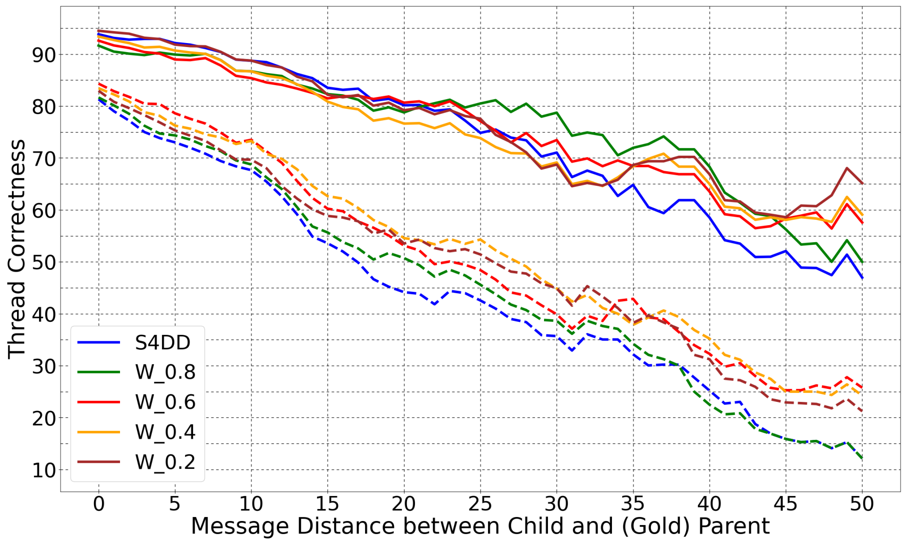


Figure 6.1: TC of SOTA and weighted loss models (shown as W_x).

6.3 Controllable Context-Based Prediction

The variations observed in the models described in Section 5.3.1 suggest that it is unlikely that in a diverse model pool, a single model outperforms the others across all contexts of a dialogue property. This motivates the need to optimally control the ensemble prediction of a model pool based on the relevant property sub-range. Existing ensembling methods for conversation disentanglement, e.g.,

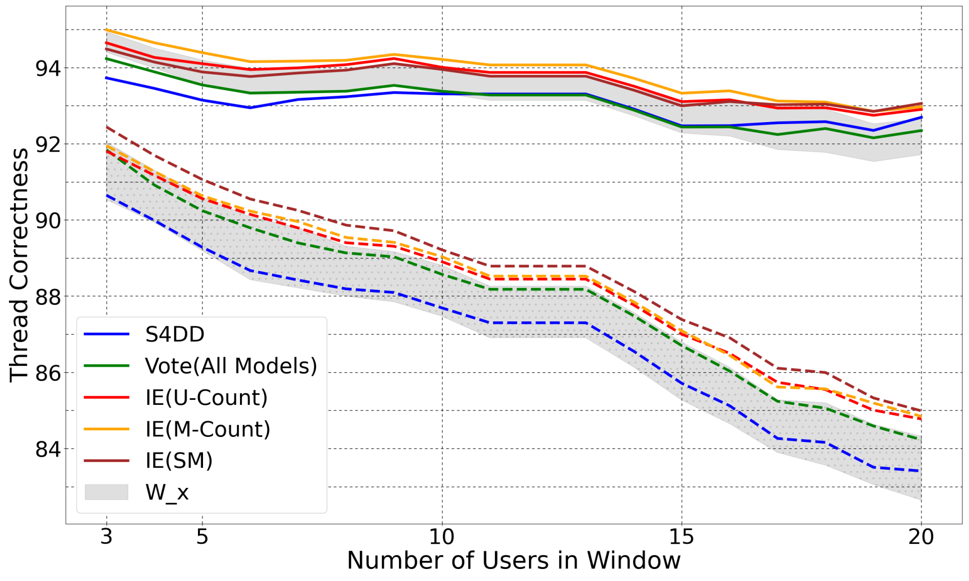


Figure 6.2: TC of SOTA, informed ensemble, and weighted loss models along user-density.

Model-AVG, Prob-AVG, and Vote-AVG by [5] and union, vote, and intersect by [2], use a fixed model pool for all predictions independent of the relevant property sub-range.

We approach this issue by proposing an informed ensembling technique, which dynamically samples a subset of models best suited for each context along a dialogue property. The model pool for our experiment contains four instances of S4DD (each trained with a different seed initialization) and four weighted loss models, i.e., $p \in \{0.2, 0.4, 0.6, 0.8\}$ as in Section 6.2. The steps involved in the ensemble are as follows:

1. **Selecting a Suitable Dialogue Property:** The informed-ensemble optimizes model selection along a single dialogue property. For this method to be applicable during inference, dialogue properties involving labeled information (e.g. actual antecedent distance) cannot be used. Thus we use the following dialogue properties:

   (a) *U-Count:* Number of users in a dialogue window (W)

   (b) *M-Count:* Number of messages by the child-user of W

   (c) *SM:* Whether the child-message contains a soft-mention

   Each property is used to create an informed ensemble, labeled as IE(`<dialogue property>`).

2. **Collecting *a priori* Model Performance:** To select the best-suited models for a context, model performances across all contexts in a dialogue property must be determined beforehand. Without access to the test-set labels, we estimate this using the TC of all models for the three dialogue properties on the dev-set of the dataset in use.

3. **Selecting Model Subset:** For each message during inference, we only use the top 10% of the best-scoring models for the relevant property sub-range.

4. **Aggregating Predictions:** Finally, the predictions of each model in the subset need to be combined to produce a single prediction. We take a majority vote of the antecedents predicted by the model subset and use the most voted-for candidate.

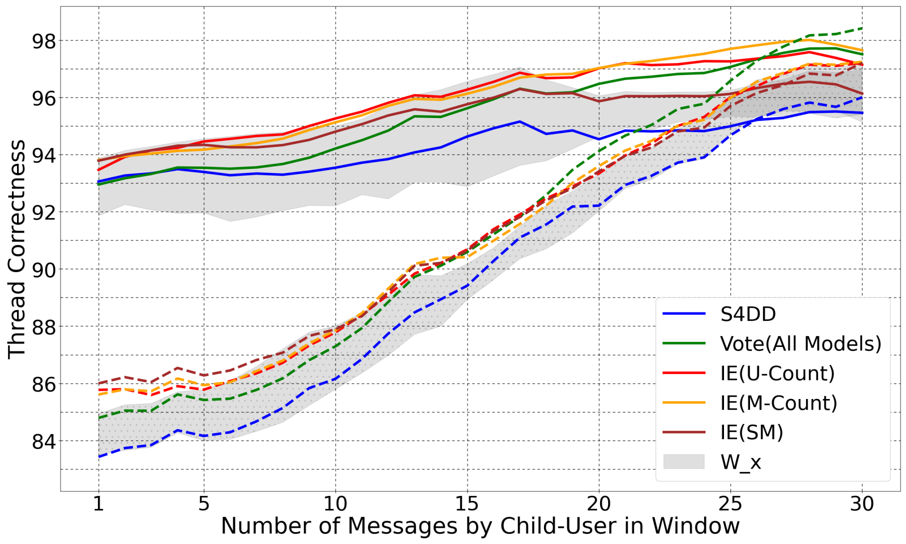


Figure 6.3: TC of SOTA, informed ensemble, and weighted loss models along user-activity.

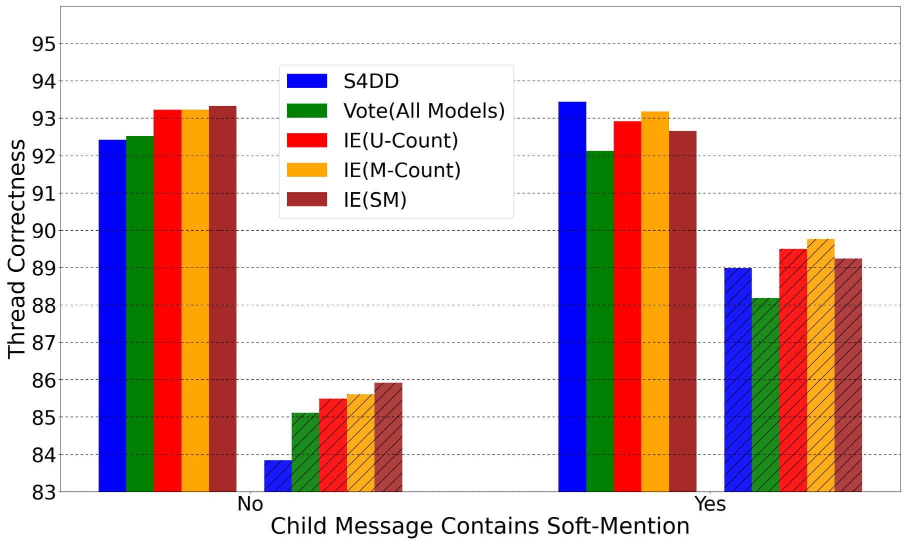


Figure 6.4: TC of SOTA, informed ensemble models along soft-mentioning.

6.3.1 Discussion

We evaluate the informed-ensembles on the IRC and IRC-Hard datasets. We also evaluate a standard majority-vote ensemble i.e. Vote(All Models), using the entire model pool to compare against our approach, which utilizes only a small subset of models for each instance. Performance on partitioning metrics is reported in Table 5.2, and the TC along the constituent dialogue properties of each informed ensemble is shown in Figures 6.2, 6.3, & 6.4. For reference, the minimum and maximum scores of all weighted-loss models across the range of U-Count & M-Count, respectively, have been added to Figure 6.2 & 6.3 (W_x in grey).

- We see that informed ensembles outperform all individual models (within the model pool) for IRC-Hard and most models for IRC.
- They also report the best TC for most contexts along U-Count & M-Count. Figure 6.4 shows that except for contexts supported by both direct and soft mentions, the ensembles also provide the best TC in SM.

NOTE: Utilizing any ensemble formulation involves an overhead of training and inferring over multiple models for each classification instance. Though this is a heavy overhead, we use this method primarily to demonstrate that leveraging the varying performance of different models can be helpful. The eventual goal is to build a unified model with apt control elements which can dynamically furnish the capabilities warranted for each dialogue context. However, up until such a model, such informed ensembling or control methods are crucial.

Chapter 7

Conclusions

In this thesis, we aim to improve accessibility to text by exploring two different NLP tasks, namely, text simplification and conversation disentanglement. Simplification aims to improve the readability of text at the sentence level, while disentanglement resolves individual conversation threads from a large multi-party dialogue. Despite differing primary objectives, both tasks convert the original format of the text into an easily comprehensible variant.

We explore text simplification in the context of a model [1], which claims to tune its simplifications along certain textual transformations (such as sequence length compression and lexical & syntactic complexity) using dedicated control tokens. We assess this claim by measuring the impact of each control token on the output text. We find that although control tokens do help in generating more accurate simplifications, thereby fetching higher scores on popular metrics, they do not always directly affect their respective text attributes. In fact, they often result in the variation of unrelated attributes or cause trivial outputs (such as the truncated version of the complex source), which adhere to the control token constraints, but do not actually result in a simplified text.

We offer two methods to address some of the above concerns: (i) An named-entity (NE) masking formulation which protects NEs from unwanted replacements and also optimizes the training vocabulary, and (ii) Using pretrained weights for input tokens to curb data sparsity induced by the added control token vocabulary and using randomly initialized weights. This sparsity, however, can be further reduced by using a more continuous solution for specifying control configurations, such as a direct scalar input to the model or at least using positional encodings for control tokens.

As part of our work on conversation disentanglement, we investigate dataset-specific features in the Ubuntu-IRC dataset [2] and asses the generalizability of models trained on IRC. Specifically, we analyze the distribution and role of direct mentions in the dataset and their resultant impact on past and proposed baseline model performance. Our in-depth qualitative analysis of the models across several key dialogue properties (such as message-antecedent distance, user density, & the presence of soft mentions) identifies points of brittleness across sub-ranges of these properties.

By leveraging the findings from the qualitative performance of all models, we provide two methods to address localized brittleness: (i) A weighted-loss formulation to better represent less frequent property sub-ranges and (ii) An informed-ensembling technique to dynamically select the ideal subset of models for each property sub-range.

Our methods outperform SOTA [3] performance on all current partitioning metrics and are more resilient across property sub-ranges. Other sources of conversation data would exhibit significantly different dialogue properties profiles. Informed ensembling across different dialogue properties should result in a more robust performance in such datasets.

We analyze the impact of the availability of direct mentions on model performance as they are predominantly artifacts of IRC and similar online platforms, thus, increasing the specificity of the dataset. A similar in-depth analysis spanning multiple features is necessary, though an investigation of this nature requires high-quality disentanglement data from other domains as well, which is currently unavailable.

In our later experiments, we apply a weighted loss formulation to tackle the poor representation of longer antecedent distances in IRC. Note that the scaling factor β can be defined using the data distribution of any dialogue property, thereby allowing a better representation of low-frequency contexts across properties.

We also show improvements in the overall model performance using an informed ensembling approach. However, it optimizes the model subset selection for only one dialogue property at a time. A more nuanced formulation that accommodates several properties simultaneously needs to be explored.

Related Publications

1. K V Aditya Srivatsa, Monil Gokani, and Manish Shrivastava. 2021. SimpleNER Sentence Simplification System for GEM 2021. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021), pages 155–160, Online. Association for Computational Linguistics*.

Bibliography

[1] Louis Martin, Éric de la Clergerie, Benoît Sagot, and Antoine Bordes. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698, Marseille, France, May 2020. European Language Resources Association.

[2] Jonathan K. Kummerfeld, Sai R. Gouravajhala, Joseph J. Peper, Vignesh Athreya, Chulaka Gunasekara, Jatin Ganhotra, Siva Sankalp Patel, Lazaros C Polymenakos, and Walter Lasecki. A large-scale corpus for conversation disentanglement. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3846–3856, Florence, Italy, July 2019. Association for Computational Linguistics.

[3] Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. Structural characterization for dialogue disentanglement. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 285–297, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[4] Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415, 2016.

[5] Tianda Li, Jia-Chen Gu, Xiaodan Zhu, Quan Liu, Zhen-Hua Ling, Zhiming Su, and Si Wei. Dialbert: A hierarchical pre-trained model for conversation disentanglement, 2020.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[7] Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679, Online, July 2020. Association for Computational Linguistics.

[8] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Compu-*

tational Linguistics (Coling 2010), pages 1353–1361, Beijing, China, August 2010. Coling 2010 Organizing Committee.

[9] Sander Wubben, Antal van den Bosch, and Emiel Krahmer. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1015–1024, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[10] William Coster and David Kauchak. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[11] Elior Sulem, Omri Abend, and Ari Rappoport. Simple and effective text simplification using semantic and neural methods. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 162–173, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[12] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning, 2017.

[13] Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. Editnts: An neural programmer-interpreter model for sentence simplification through explicit editing, 2019.

[14] Katja Filippova. Controlled hallucinations: Learning to generate faithfully from noisy data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870, Online, November 2020. Association for Computational Linguistics.

[15] Angela Fan, David Grangier, and Michael Auli. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[16] Louis Martin, Samuel Humeau, Pierre-Emmanuel Mazaré, Antoine Bordes, Éric Villemonte de la Clergerie, and Benoît Sagot. Reference-less quality estimation of text simplification systems. *CoRR*, abs/1901.10746, 2019.

[17] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.

[18] Gustavo Paetzold and Lucia Specia. Semeval 2016 task 11: Complex word identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 560–569, 2016.

[19] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python, 2020.

[20] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[21] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[22] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[24] Juntao Yu, Bernd Bohnet, and Massimo Poesio. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online, July 2020. Association for Computational Linguistics.

[25] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[26] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[27] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[28] Micha Elsner and Eugene Charniak. You talking to me? a corpus and algorithm for conversation disentanglement. In *Proceedings of ACL-08: HLT*, pages 834–842, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[29] Micha Elsner and Warren Schudy. Bounding and comparing methods for correlation clustering beyond ILP. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 19–27, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[30] Micha Elsner and Eugene Charniak. Disentangling chat. *Computational Linguistics*, 36(3):389–409, September 2010.

[31] Micha Elsner and Eugene Charniak. Disentangling chat with local coherence models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1179–1189, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[32] Lidan Wang and Douglas W. Oard. Context-based message expansion for disentanglement of interleaved text conversations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 200–208, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[33] Shikib Mehri and Giuseppe Carenini. Chat disentanglement: Identifying semantic reply relationships with random forests and recurrent neural networks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 615–623, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[34] Jyun-Yu Jiang, Francine Chen, Yan-Ying Chen, and Wei Wang. 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)*, pages 1812–1822, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[35] David Uthus and David Aha. Detecting bot-answerable questions in Ubuntu chat. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 747–752, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing.

[36] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic, September 2015. Association for Computational Linguistics.

[37] Amy Zhang, Bryan Culbertson, and Praveen Paritosh. Characterizing online discussion using coarse discourse sequences. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):357–366, May 2017.

[38] Henghui Zhu, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. Who did they respond to? conversation structure modeling using masked hierarchical transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9741–9748, Apr. 2020.

[39] Preetha Chatterjee, Kostadin Damevski, Nicholas A. Kraft, and Lori Pollock. Software-related slack chats with disentangled conversations. In *Proceedings of the 17th International Conference on Mining Software Repositories*, MSR '20, page 588–592, New York, NY, USA, 2020. Association for Computing Machinery.

[40] Keerthana Muthu Subash, Lakshmi Prasanna Kumar, Sri Lakshmi Vadlamani, Preetha Chatterjee, and Olga Baysal. Disco: A dataset of discord chat conversations for software engineering research. In *Proceedings of the 19th International Conference on Mining Software Repositories*, MSR '22, page 227–231, New York, NY, USA, 2022. Association for Computing Machinery.

[41] Hui Liu, Zhan Shi, Jia-Chen Gu, Quan Liu, Si Wei, and Xiaodan Zhu. End-to-end transition-based online dialogue disentanglement. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3868–3874, Online, 7 2020. International Joint Conferences on Artificial Intelligence Organization. Main track.

[42] Lin Shi, Xiao Chen, Ye Yang, Hanzhi Jiang, Ziyou Jiang, Nan Niu, and Qing Wang. A first look at developers' live chat on gitter. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 391–403, New York, NY, USA, 2021. Association for Computing Machinery.

[43] Tao Yu and Shafiq Joty. Online conversation disentanglement with pointer networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6321–6330, Online, November 2020. Association for Computational Linguistics.

[44] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28, Online, 2015. Curran Associates, Inc.

[45] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

[47] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[48] Jia-Chen Gu, Chongyang Tao, Zhenhua Ling, Can Xu, Xiubo Geng, and Daxin Jiang. MPC-BERT: A pre-trained language model for multi-party conversation understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3682–3692, Online, August 2021. Association for Computational Linguistics.

[49] Ozan İrsoy, Rakesh Gosangi, Haimin Zhang, Mu-Hsin Wei, Peter Lund, Duccio Pappadopulo, Brendan Fahy, Neophytos Nephytou, and Camilo Ortiz. Dialogue act classification in group chats with dag-lstms, 2019.

[50] Duccio Pappadopulo, Lisa Bauer, Marco Farina, Ozan İrsoy, and Mohit Bansal. Disentangling online chats with DAG-structured LSTMs. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 152–159, Online, August 2021. Association for Computational Linguistics.

[51] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[52] Marina Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007.

[53] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985.

[54] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[55] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[56] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural*

Language Processing: System Demonstrations, pages 38–45, Online, October 2020. Association for Computational Linguistics.