ProoFVer: Natural Logic Theorem Proving for Fact Verification

Amrith Krishna
Department of Computer
Science, University of
Cambridge, United Kingdom
ak2329@cam.ac.uk

Sebastian Riedel
Meta AI and University College
London, United Kingdom
sriedel@fb.com

Andreas Vlachos
Department of Computer
Science, University of
Cambridge, United Kingdom
av308@cam.ac.uk

Abstract

Fact verification systems typically rely on neural network classifiers for veracity prediction, which lack explainability. This paper proposes ProoFVer, which uses a seq2seq model to generate natural logic-based inferences as proofs. These proofs consist of lexical mutations between spans in the claim and the evidence retrieved, each marked with a natural logic operator. Claim veracity is determined solely based on the sequence of these operators. Hence, these proofs are faithful explanations, and this makes ProoFVer faithful by construction. Currently, ProoFVer has the highest label accuracy and the second best score in the FEVER leaderboard. Furthermore, it improves by 13.21% points over the next best model on a dataset with counterfactual instances, demonstrating its robustness. As explanations, the proofs show better overlap with human rationales than attention-based highlights and the proofs help humans predict model decisions correctly more often than using the evidence directly.[1]

1 Introduction

Fact verification systems typically comprise an evidence retrieval model followed by a textual entailment classifier (Thorne et al., 2018b). Recent high-performing fact verification systems (Zhong et al., 2020; Ye et al., 2020) use neural models for textual entailment whose reasoning is opaque to humans despite advances in interpretablity (Han et al., 2020). On the other hand, proof systems like NaturalLI (Angeli and Manning, 2014) provide transparency in their decision making for entailment tasks, by using explicit proofs in the form of natural logic. However, the accuracy of such approaches often does not match that of neural models (Abzianidze, 2017a).

Justifying decisions is central to fact verification (Uscinski and Butler, 2013). While models such as those developed for FEVER (Thorne et al., 2018b) typically substantiate their decisions by presenting the evidence as is, more recent proposals use the evidence to generate explanations. Here, models highlight salient parts of the evidence (Popat et al., 2018; Wu et al., 2020), generate summaries (Kotonya and Toni, 2020b; Atanasova et al., 2020), correct factual errors (Thorne and Vlachos, 2021b; Schuster et al., 2021), answer claim-related questions (Fan et al., 2020), or perform rule discovery (Ahmadi et al., 2019; Gad-Elrab et al., 2019). An explanation is faithful only if it reflects the information that is used for decision making (Lipton, 2018; Jacovi and Goldberg, 2020), which these systems do not guarantee. A possible exception here would be the rule discovery models, although, their performance often suffers due to limited knowledge base coverage and/or the noise in rule extraction from text (Kotonya and Toni, 2020a; Pezeshkpour et al., 2020). Faithful explanations are useful as mechanisms to dispute, debug, or advise (Jacovi and Goldberg, 2021), which may aid a news agency for advice, a user to dispute decisions, and a developer for model debugging in fact verification.

Keeping both accuracy and explainability in mind, we propose **ProoFVer**—**Proof** System for **F**act **Ver**ification—which generates proofs or refutations of the claim given evidence as natural logic-based inference. ProoFVer follows the natural logic based theory of compositional entailment, originally proposed in NatLog (MacCartney and Manning, 2007). In the example of Figure 1 ProoFVer generates the proof shown in Figure 2, for a given claim and evidence. Here, at each step in the proof, a claim span is mutated with a span from the evidence. Each such mutation is marked with an entailment relation, by assigning a natural logic operator (NatOp; Angeli and

[1] Find our code and data at https://github.com/krishnamrith12/ProoFVer.

Action Editor: Mohit Bansal. Submission batch: 10/2021; Revision batch: 6/2022; Published 9/2022.

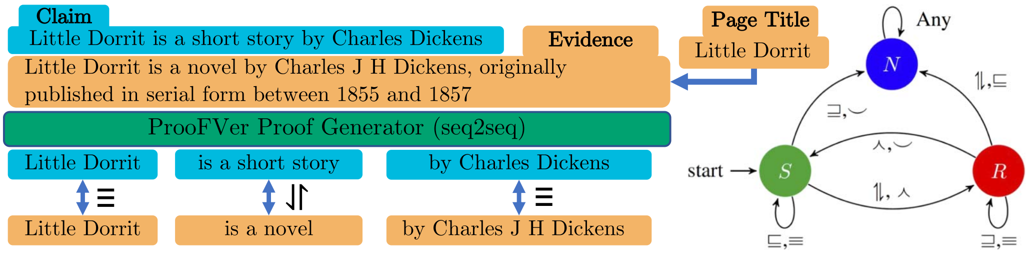
Claim
Little Dorrit is a short story by Charles Dickens
Evidence
Little Dorrit is a novel by Charles J H Dickens, originally published in serial form between 1855 and 1857
Page Title
Little Dorrit
ProoFVer Proof Generator (seq2seq)
Little Dorrit
is a short story
by Charles Dickens
≡
⥮
≡
Little Dorrit
is a novel
by Charles J H Dickens
Any
N
⥮,⊑
⊒,⌣
∧,⌣
start
S
R
⥮, ∧
⊑,≡
⊒,≡

Figure 1: The proof generator in ProoFVer, generates the natural logic proofs using a seq2seq model. The natural logic operators from the proof are used as transitions in the DFA to determine the veracity of the claim. The states S, R, and N in the automaton denote the task labels Supports, Refutes, and Not Enough Info, respectively. The transitions in the automaton are the natural logic operators (NatOPs) defined in Table 1.

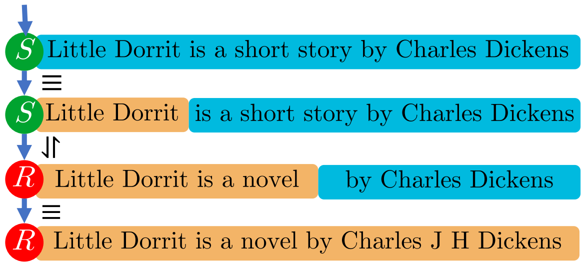
S
Little Dorrit is a short story by Charles Dickens
≡
S
Little Dorrit
is a short story by Charles Dickens
⥮
R
Little Dorrit is a novel
by Charles Dickens
≡
R
Little Dorrit is a novel by Charles J H Dickens

Figure 2: Proof steps for the input in Figure 1.

Manning, 2014). A step in the proof can be represented using a triple, consisting of the aligned spans in the mutation and its assigned NatOp. In the example, the mutations in the first and last triples occur with semantically equivalent spans, and hence are assigned with the equivalence NatOp (≡). However, the mutation in the second triple results in a contradiction, as 'short story' is replaced with 'novel' and an item cannot be both. Hence, the mutation is assigned the alternation NatOp (⥮). The sequence of NatOps from the proof become the transitions in the DFA shown in Figure 1, which in this case terminates at the 'Refute (R)' state, that is, the evidence refutes the claim.

Unlike other natural logic systems (Angeli et al., 2016; Feng et al., 2020), ProoFVer can form a proof by combining spans from multiple evidence sentences, by leveraging the entity mentions linking those sentences. The proof is generated by a seq2seq model trained using a heuristically annotated dataset, obtained by combining information from the publicly available FEVER dataset (Thorne et al., 2018a; Thorne and Vlachos, 2021b) with PPDB (Pavlick et al., 2015), Wordnet (Miller, 1995) and Wikidata (Vrandečić and Krötzsch, 2014). We heuristically generate the training data for the claims in three datasets, namely, FEVER, symmetric FEVER (Schuster et al., 2019), and FEVER 2.0 (Thorne et al., 2019).

ProoFVer is currently the highest scoring system on the FEVER leaderboard in terms of label accuracy and is the second-best system in terms of FEVER score. Additionally, ProoFVer has robustness and explainability as its key strengths. Its veracity predictions are solely determined using the generated proof. Hence by design, ProoFVer's proofs, when used as explanations, are faithful by construction (Lei et al., 2016; Jain et al., 2020). Similarly, it demonstrates robustness to counterfactual instances from Symmetric FEVER and adversarial instances from FEVER 2.0. In particular, ProoFVer achieved 13.21% higher label accuracy than that of the next best model (Ye et al., 2020) for symmetric FEVER and similarly improves upon the previous best results (Schuster et al., 2021) on Adversarial FEVER.

To evaluate the robustness of fact verification systems against the impact of superfluous information from the retriever, we propose a new metric, Stability Error Rate (SER), which measures the proportion of instances where superfluous information changes the decision of the model. ProoFVer achieves a SER of 5.73%, compared with 9.36% of Stammbach (2021), where a lower SER is preferred. ProoFVer's proofs as explanations, apart from being faithful, score high in their overlap with human rationales with a token overlap F1-Score of 93.28%, 5.67 percentage points more than attention-based highlights from Ye et al. (2020). Finally, humans, with no knowledge of natural logic, correctly predict ProoFVer's decisions 81.67% of the times compared with 69.44% when using the retrieved evidence.

2 Natural Logic Proofs as Explanations

Natural logic operates directly on natural language (Angeli and Manning, 2014; Abzianidze, 2017b). Thus it is appealing for fact verification, as structured knowledge bases like Wikidata typically lag behind text-based encyclopedias such as Wikipedia in terms of coverage (Johnson, 2020). Furthermore, it obviates the need to translate claims and evidence into meaning representations such as lambda calculus (Zettlemoyer and Collins, 2005). While such representations may be more expressive, they require the development of semantic parsers, introducing another source of potential errors in the verification process.

Natural Logic has been previously used in several information extraction and NLU tasks such as Natural Language Inference (NLI, Abzianidze, 2017a; Feng et al., 2020), question answering (Angeli et al., 2016), and open information extraction (Angeli, 2016, Chapter 5). NatLog (MacCartney and Manning, 2007), building on earlier theoretical work on natural logic and monotonicity calculus (Van Benthem, 1986; Valencia, 1991), uses natural logic for textual inference.

NaturalLI (Angeli and Manning, 2014) extended NatLog by adopting the formal semantics of Icard III and Moss (2014), and it is a proof system formulated for the NLI task. It determines the entailment of a hypothesis by searching over a database of premises. The proofs are in the form of a natural logic based logical inference, which results in a sequence of mutations between a premise and a hypothesis. Each mutation is marked with a natural logic relation, and is realized as a lexical substitution, forming a step in the inference. Each mutation results in a new sentence, and the natural logic relation assigned to it identifies the type of entailment that holds between the sentences before and after the mutation. NaturalLI adopts a set of seven natural logic operators, as shown in Table 1. The operators were originally proposed in NatLog (MacCartney, 2009, p. 79). We henceforth refer to these operators as *NatOps*.

To determine whether a hypothesis is entailed by a premise, NaturalLI uses a deterministic finite state automaton (DFA). Here, each state is an entailment label, and the transitions are the NatOps (Figure 1). The sequence of NatOps in the inference is used to traverse the DFA, and the state where it terminates decides the label of the hypothesis-premise pair. The decision making process relies solely on the steps in the logical inference, and thus form faithful explanations.

NatOP: Name	Definition
⥮: Alternation	$x \cap y = \emptyset \wedge x \cup y \neq U$
⌣: Cover	$x \cap y \neq \emptyset \wedge x \cup y = U$
≡: Equivalence	$x = y$
⊑: Forward Entailment	$x \subset y$
⅄: Negation	$x \cap y = \emptyset \wedge x \cup y = U$
⊒: Reverse Entailment	$x \supset y$
#: Independence	All other cases

Table 1: Natural logic relations (NatOps) and their set theoretic definitions.

Other proof systems that apply mutations between text sequences have been previously explored. Stern et al. (2012) explored how to transform a premise into a hypothesis using mutations, however their approach was limited to two-way entailment instead of three-way that is handled by NaturalLI. Similar proof systems have used mutations in the form of tree-edit operations (Mehdad, 2009), transformations over syntactic parses (Heilman and Smith, 2010; Harmeling, 2009), knowledge-based transformations in the form of lexical mutations, entailment rules, rewrite rules, or their combinations (Bar-Haim et al. 2007; Szpektor et al., 2004).

3 ProoFVer

ProoFVer uses a seq2seq generator that generates a proof in the form of natural-logic based logical inference, which becomes the input to a DFA for predicting the veracity of the claim. We elaborate on the proof generation process in Section 3.1, and on the veracity prediction in Section 3.2.

3.1 Proof Generation

The proof generator, as shown in Figures 1 and 3, takes as input a claim along with one or more retrieved evidence sentences. It generates the steps of the proof as a sequence of triples, each consisting of a span from the claim, a span from the evidence and a NatOp. The claim span being substituted and the evidence span replacing it form a *mutation*, and each mutation is assigned a NatOp. In a proof, we start with the claim, and the mutations are iteratively applied from left to right. Figure 1 shows a proof containing a sequence

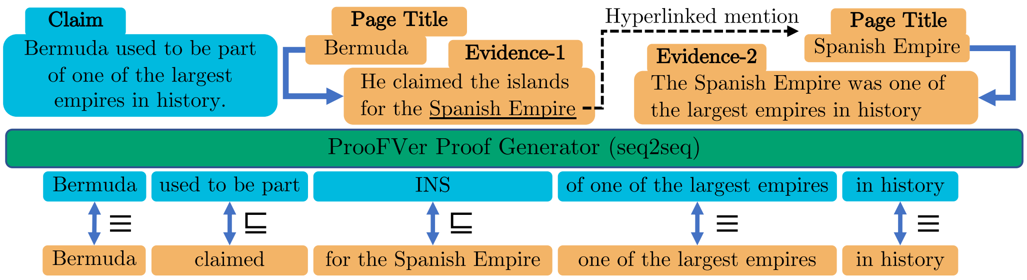
Claim
Bermuda used to be part of one of the largest empires in history.
Page Title
Bermuda
Evidence-1
He claimed the islands for the Spanish Empire
Hyperlinked mention
Page Title
Spanish Empire
Evidence-2
The Spanish Empire was one of the largest empires in history
ProoFVer Proof Generator (seq2seq)
Bermuda
used to be part
INS
of one of the largest empires
in history
≡
⊑
⊑
≡
≡
Bermuda
claimed
for the Spanish Empire
one of the largest empires
in history

Figure 3: A claim requiring multiple evidence sentences for verification.

of three triples. The corresponding mutated statements at each step of the proof, along with the assigned NatOps, are shown in Figure 2.

We use a seq2seq model following an autoregressive formulation for the proof generation. In the proof, successive spans of the claim form part of the successive triples. However, the corresponding evidence spans in the successive triples need not follow any order. As shown in Figure 3, the evidence spans may come from multiple sentences, and may not all end up being used. Finally, the NatOps, as shown in Table 1, are represented using a predetermined set of tokens.

To obtain valid proofs during prediction, we need to lexically constrain the inference process by switching between three different search spaces depending on which element of the triple is being predicted. To achieve this, we use dynamically constrained markup decoding (De Cao et al., 2021), a modified form of lexically constrained decoding (Post and Vilar, 2018). This decoding uses markups to switch between the search spaces, and we use the delimiters ``{'', ``}'', ``['', and ``]'' as the markups. Using these markups, we constrain the tokens predicted between a ``{'' and ``}'' to be from the claim, between a ``['', and ``]'' to be from the evidence, and the token after ``]'' to be a NatOp token. The prediction of a triple begins with predicting a ``{'', and it proceeds by generating a claim span where the tokens are monotonically copied from the claim in the input, until a ``}'' is predicted. The prediction then continues by generating a ``['', which initiates the evidence span prediction in the triple. The evidence span can begin with any word from the evidence, and is then expanded by predicting subsequent tokens, until ``]'' is predicted. Finally, the NatOp token is predicted. In the next triple, copying resumes from the next token in the claim. All triples until the one with the last token in the claim are generated in this manner.

3.2 Veracity Prediction

The DFA shown in Figure 1 uses the sequence of NatOps predicted by the proof generator as transitions to arrive at the outcome. Figure 2 shows the corresponding sequence of transitions for the claim and evidence from Figure 1. Based on this, the DFA in Figure 1 determines that the evidence refutes the claim, that is, it terminates in state R. NaturalLI (Angeli and Manning, 2014) designed the DFA for the three classes in the NLI classification task, namely, entail, contradict, and neutral. Here, we replace them with SUPPORT (S), REFUTE (R), and NOT ENOUGH INFO (N), respectively for fact verification. Angeli and Manning (2014) chose not to distinguish between negation (⋏) and alternation (⥮) relations for NLI, and assign ⥮ for both. However, there is a clear distinction between cases where each of these NatOPs is applicable in fact verification, and thus we treat them as different NatOps. For instance, in the second mutation for the claim in Figure 1, an evidence span ``is not a short story'' would be assigned negation (⋏), and not the currently assigned alternation (⥮) for the mutation with the evidence span ``is a novel''. However, we follow Angeli and Manning (2014) in not using the cover (⌣) NatOp. In rare occasions where this NatOp would be applicable, say in a mutation with the spans ``not a novel'' and ``fiction'', we currently assign the independence NatOp (⋕).

4 Generating Proofs for Training

Training datasets for evidence-based fact verification consist of instances containing a claim, a label indicating its veracity, and the evidence, typically a set of sentences (Thorne et al., 2018a; Hanselowski et al., 2019; Wadden et al., 2020). However, we need sequences of triples to train the proof generator of Section 3.1. Manually

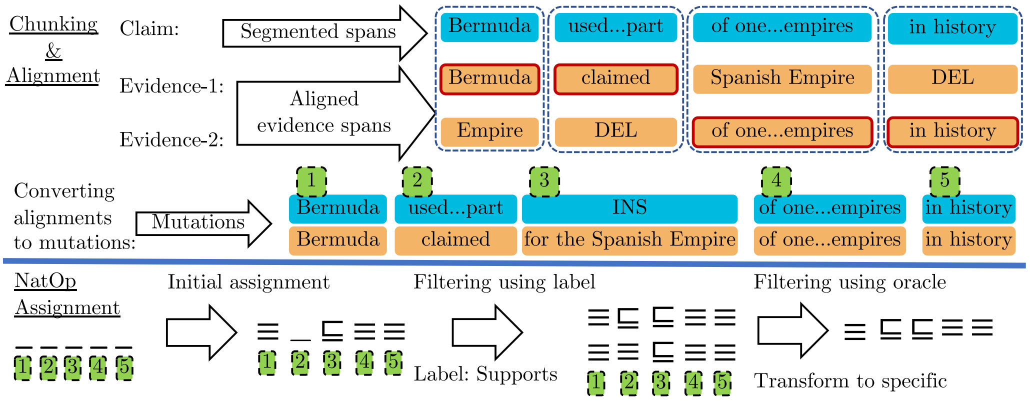

Figure 4: Annotation process for obtaining the proof for the input in Figure 3. It proceeds in two steps, chunking & alignment, and NatOp assignment, and the latter proceeds by initial mutation assignment and two filtering steps.

annotating them would be laborious; thus, we heuristically generate them from existing resources. As shown in Figure 4, we perform a two-step annotation process: chunking and alignment, followed by the NatOp assignment.

4.1 Chunking and Alignment

Chunking the claim into spans is conducted using the chunker of Akbik et al. (2019), and any span that does not contain any content words is merged with its subsequent span. Next, as shown in Figure 4, a word aligner (Jalili Sabet et al., 2020) aligns each evidence sentence in the input separately with the claim. For each claim span, each evidence sentence provides an aligned span by grouping together words that are aligned to it, including any words in between to ensure contiguity. However, if the aggregated similarity score from the aligner for a given pair of claim and evidence spans falls below an empirically set threshold, then it is ignored and instead the claim span is aligned with the string ‘‘DEL’’. In Figure 4, ‘‘DEL’’ appears once in each of the evidence sentences.

Next, we convert the alignments into a sequence of mutations, which requires no additional effort in instances with only one evidence sentence. However, a claim span may have multiple evidence spans aligned with it in cases with multiple evidence sentences, as shown in Figure 4. Here, for a claim span, we generally select the evidence span with the highest cosine similarity with it. Such spans are marked with solid red borders in Figure 4. Further, we assume that the evidence sentences are linked via entity mentions, such as ‘‘Spanish Empire’’ the only hyperlinked mention (from Evidence-1 to 2) in Figure 3. These hyperlinked mentions must always be added as a mutation, as they provide the context for switching the source of the evidence from one sentence to another. In Figure 3, ‘‘Spanish Empire’’ is not selected as an alignment based on the similarity scores with the claim spans. Hence, it is inserted as the third mutation, at the juncture at which the switch from Evidence-1 to 2 happens. It is aligned with the string ‘‘INS’ in the place of a claim span. Use of hyperlink structure in Wikipedia or performing entity linking to establish hyperlinked mentions, similar to our approach here, has been previously explored in multi-hop open domain question answering (Asai et al., 2020; Nie et al., 2019). Mutations with a ‘‘DEL’’ instead of an evidence span, and an ‘‘INS’’ instead of a claim span, are treated as deletions and insertions of claim and evidence spans, respectively.

4.2 NatOp Assignment

As shown in Figure 4, the NatOp assignment step produces a sequence of NatOps, one for each mutation. Here, the search space becomes exponentially large (i.e., 6^n possible NatOp sequences for n mutations). First, we assign NatOps to individual mutations relying on hand-crafted rules and external resources, without considering the other mutations in the sequence (§ 4.2.1). With this partially filled NatOp sequence, we perform two filtering steps to further reduce the search space. We describe these steps below: one using veracity label information from training data in FEVER (Thorne et al., 2018a) and another using some additional manual annotation information from annotation logs of FEVER (§ 4.2.2).

4.2.1 Initial Assignment

The initial assignment of NatOps considers each mutation in the sequence in isolation. Here, mutations that fully match lexically are assigned with the equivalence NatOp ($\equiv$), like the mutations 1, 4, and 5 in Figure 4. Similarly, mutations where the claim or evidence span has an extra negation word but lexically match otherwise, are assigned the negation NatOp ($\curlywedge$). Further, insertions and deletions, that is, mutations with Ins and Del, respectively (§4.1), containing negation words are also assigned the negation NatOp. To obtain these words, we identify a set of common negation words from the list of stop words in Honnibal et al. (2020), and combine them with the list of negative sentiment polarity words from Hu and Liu (2004). Remaining cases of insertions (deletions) are treated as making the existing claim more specific (general), and hence assigned the forward (reverse) entailment NatOp, like mutation 3 in Figure 4. Furthermore, as every paraphrase pair present in Paraphrase Database (PPDB Ganitkevitch et al., 2013; Pavlick et al., 2015) is marked with an entailment relation, we identify mutations which are present in it as paraphrases and assign the corresponding NatOp.

In several cases, the NatOp information need not be readily available at the span level. Here, we retain the word-level alignments from the aligner and perform lexical level NatOp assignment with the help of Wordnet (Miller, 1995) and Wikidata (Vrandečić and Krötzsch, 2014). We follow MacCartney (2009, Chapter 6) for NatOp assignment of open-class terms using Wordnet.

Additionally, we define rules to assign a NatOp for named entities using Wikidata. Here, aliases of an entity are marked with an equivalence NatOp ($\equiv$), as shown in third triple in Figure 1. Further, we manually assign NatOps to the 500 most frequently occurring Wikidata relations in the aligned training data. For instance, as shown in Figure 5, the entities 'The Trial' and 'novel' have the relation 'genre'. A claim span containing 'The Trial', when substituted with an evidence span containing 'novel', would result in a generalisation of the claim, and hence will be assigned the reverse entailment NatOp ($\sqsupseteq$). A substitution in the reverse direction would be assigned a forward entailment NatOp ($\sqsubseteq$), indicating specialization.

The KB relations we annotated occur between the entities linked in Wikidata, and they do not capture hierarchical multihop relations between the entities in the KB. We create such a hierarchy by combining the ''instance of'', ''part of'', and ''subclass of'' relations in Wikidata. Thus, a pair of entities connected via a directed path of length $k \leq 3$, such as ''Work of art'' and ''Rashomon'' in Figure 5, is considered to have a parent-child relation, and assigned the forward or reverse entailment NatOp, depending on which span appears in the claim and the evidence. Similarly, two entities (e.g., ''Rashomon'' and ''Inception'') are considered to be siblings if they have a common parent, and are assigned the alternation NatOp ($\upharpoonleft\!\downharpoonright$). However, two connected entities that do not satisfy the aforementioned distance criterion (e.g., ''novel'' and ''Rashomon'') are assigned with the independence NatOp (#), signifying that they are unrelated.

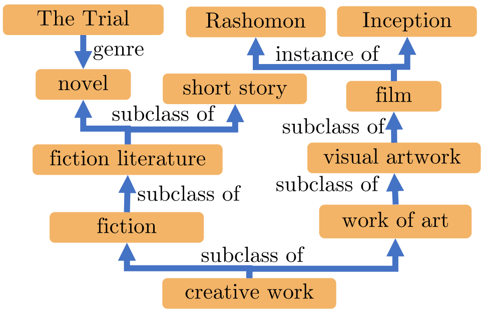

Figure 5: Entities and their relations in Wikidata.

4.2.2 Filtering the Search Space

While in Section 4.2.1 we assigned a NatOp to each mutation in isolation, there can still be unfilled NatOps. For instance, the unfilled NatOp in the second mutation of Figure 4 leads to six possible NatOp sequences as candidates, one per available NatOp. Recall that these NatOp sequences act as a transition sequence in the DFA (§ 3.2). Thus we make use of the partially filled NatOp sequence and the veracity label from the training data to filter out NatOp sequences that do not terminate at the same state as the veracity label according to the DFA. The instance in Figure 4 has the Support label, and among the six possible candidate sequences only two terminate in this label. Hence, we retain those two sequences.

For the final filtering step we use the additional manual annotation that was produced during the construction of the claims in FEVER. There, the annotators constructed each claim by manipulating a factoid extracted from Wikipedia using

Transformation	S	R	N
substitute with similar info.	⊑	⥮	⊒
substitute with dissimilar info.	⊑	⥮	⋕
paraphrasing	≡	⥮	⋕
negation	⋏	⋏	⋏
transform to specific	⊑	⊑	⊑
transform to general	⊒	⊒	⊒

Table 2: NatOp assignment based on transformations and veracity label information.

one of the six transformations listed in Table 2. Our proofs can be viewed as an attempt at reconstructing the factoid from a claim in multiple mutations, whereas these transformations can be considered claim-level mutations that transition directly from the last step (reconstructed factoid) in the proof to the first step (claim). This factoid is treated as the corrected claim in Thorne and Vlachos (2021b), who released this annotation. For each veracity label we define the mapping of each transformation to a NatOp, as described in Table 2. The assumption is that if a transformation has resulted in a particular veracity label, then the corresponding NatOp is likely to occur in the proof. To identify the mutation to assign it, we obtain the text portions in the claim manipulated by the annotators to construct it, by comparing the claim and the original Wikipedia factoid. In the example of Figure 4, this transformed text span happens to be part of the second mutation, and as per Table 2 forward entailment is the corresponding NatOp given the veracity label, resulting in the selection of the first NatOp sequence. In rare occasions (2.55% claims in FEVER), we manually performed NatOp assignment, as the filtering steps led to zero candidates in those cases. As the heuristic annotation requires manual effort, we explore how it can be obtained using a supervised classifier (see §5.5).

5 Experimental Methodology

5.1 Data

ProoFVer is trained using heuristically annotated proofs (§4) obtained from FEVER (Thorne et al., 2018a), which has a train/test/development split of 145,449/19,998/19,998 claims. Further, the heuristic proof annotation involves the use of additional information from the manual annotation logs of FEVER, recently released by Thorne and Vlachos (2021b). Finally, claims with the label Not Enough Info (NEI) require retrieved evidence for obtaining their proofs for training, as no ground truth evidence exists for such cases. Here, we use the same retriever that would be used during the prediction time as well.

In addition to FEVER, we train and evaluate ProoFVer on two other related datasets. First, we use Symmetric FEVER (Schuster et al., 2019), a dataset designed to assess the robustness of fact verification systems against the claim-only bias present in FEVER. The dataset consists of 1,420 counterfactual instances, split into development and test sets of 708 and 712 instances, respectively. Here, we heuristically generate the ground truth proofs for the dataset's development data and use it to fine tune ProoFVer, before evaluating it on the dataset's test data. Similarly, we also evaluate ProoFVer on the FEVER 2.0 adversarial examples (Thorne et al., 2019). Specifically, we use the same evaluation subset of 766 claims that was used by Schuster et al. (2021). To fine-tune ProoFVer on this dataset, we generate the ground truth proofs for 2,100 additional adversarial claims, separate from the evaluation set, which were curated by the organisers and participants of the FEVER 2.0 shared task.

Finally, we also use the manual annotation logs of FEVER (Thorne and Vlachos, 2021b) to obtain rationales for claims in the development data. In particular, we obtain the rationale for a claim by extracting from its corresponding Wikipedia factoid the words which were removed by the annotators during its creation. If these words are part of an evidence sentence, then they become the rationale for veracity label of the claim given the evidence. Further, we require that the words extracted as rationale form a contiguous phrase. We identified 300 claims that satisfy all these criteria.

5.2 Evaluation Metrics

The evaluation metrics for FEVER are label accuracy (LA, i.e., veracity accuracy) and FEVER Score (Thorne et al., 2018b), which rewards only those predictions which are accompanied by at least one correct set of evidence sentences. We report mean LA and standard deviation for experiments with Symmetric FEVER, where we use its development data for training and train with five random initialisations due to its limited size. We further introduce a new evaluation metric, to assess model robustness, called Stability Error Rate (SER). Neural models, especially with a retriever component, have shown to be vulnerable to model overstability (Jia and Liang, 2017). Overstability is the inability of a model to distinguish superfluous information that merely has lexical similarity with the input, from the information truly relevant to arrive at the correct decision. In the context of fact verification, it is expected that an ideal model should always predict NOT ENOUGH INFO, whenever it lacks sufficient evidence to make a decision otherwise. Further, it should arrive at a REFUTE or SUPPORT decision only when the model possesses sufficient evidence to do so, and any additional evidence should not alter its decision. To assess the model overstability in fact verification, we define SER as the percentage of claims where additional evidence alters the SUPPORT or REFUTE decision of a model.

5.3 Baseline Systems

KGAT (Liu et al., 2020) uses a graph attention network, where each evidence sentence, concatenated with the claim, forms a node in the graph. We use their best configuration, where the node representations are initialized using RoBERTA (Large). The relative importance of each node is computed with node kernels, and information propagation is performed using edge kernels. They also propose a new evidence sentence retriever, a BERT model trained with a pairwise ranking loss, though they rely on past work for document retrieval (Hanselowski et al., 2018).

CorefBERT (Ye et al., 2020) follows KGAT and differs only in terms of the LM used for the node initialisation. Here, they further pretrain the LM on a task that involves prediction of referents of a masked mention to capture co-referential relations in context. We use CorefRoBERTA, their best-performing configuration.

DominikS (Stammbach, 2021) focuses primarily on sentence-level evidence retrieval, scoring individual tokens from a given Wikipedia document, and then selecting the highest scoring sentences by averaging token scores. It uses a fine-tuned document level BigBird model (Zaheer et al., 2020) for this purpose. For claim verification it uses a DeBERTa (He et al., 2021) based classifier.

5.4 ProoFVer: Implementation Details

We follow most previous works on FEVER which model the task in three steps, namely, document retrieval, retrieval of evidence sentences from them, and finally veracity prediction based on the evidence. ProoFVer's novelty lies in the proof generation in the third step. Hence, for better comparability, we follow two popular, well-performing retrieval approaches, Liu et al. (2020) and Stammbach (2021). Liu et al.'s (2020) sentence retriever, also used in Ye et al. (2020), is a sentence level pairwise ranking model, whereas that of Stammbach (2021) is a document level token score aggregation model. ProoFVer's configuration which uses the former is our default configuration, referred to as ProoFVer, and the configuration using the latter will henceforth be referred to as ProoFVer-SB. We retrieve five sentences for each claim as required in the FEVER evaluation.

For the proof generator, we use the pretrained BART (Large) model (Lewis et al., 2020) and fine-tune it using the heuristically annotated data from Section 4. During prediction, the search spaces for the claim and evidence are populated using two separate tries. We add all possible subsequences of the claim and evidence, each with one to seven words, into the respective tries. The default configuration takes the concatenation of a claim and all the retrieved evidence together as a single input, separated by a delimiter.

We consider three additional configurations which differ in the way the retrieved evidence is handled. In ProoFVer-MV, a claim is concatenated with one evidence sentence at a time; this produces five proofs and five decisions per claim, and the final label is decided based on majority voting (MV). Both ProoFVer-A and -AR are designed to restrict the proof generator's flexibility in inferring the textual spans in the mutations, and thus assess the gains obtained by allowing it in ProoFVer. ProoFVer-A (aligned) considers during prediction only the subsequences from each evidence sentence aligned with the claim using word-level alignment, which are then concatenated with the claim as its input during training and prediction. Thus, the evidence search space becomes narrower, as the unaligned portions in the evidence are not considered. ProoFVer-AR (aligned-restricted) further restricts the search space of both the claim and evidence, by predetermining the number of mutations, the claim spans in these mutations and five candidate evidence spans for each mutation (one per evidence sentence). It obtains this information using the chunker and aligner used in the heuristic annotation (§4).

5.5 Heuristic Annotation Using Kepler

To reduce the reliance on manual annotation from Thorne and Vlachos (2021b) during the annotation in Section 4, we experiment with replacing the ground truth transformations with predicted ones using a classifier. We use KEPLER (Wang et al., 2021), a RoBERTA-based pretrained LM enhanced with KB relations and entity pairs from WikiData for the classification. KEPLER covers 97.5% of the entities present in FEVER. We first train it with the FEVER training dataset for the fact verification task. Then we fine-tune it for the six-class classification task of predicting the transformations, given a claim, evidence sentence and veracity label as input from the FEVER training data. We train it with varying training dataset sizes ranging from 1.24% (1,800; 300 per class) to 41.24% (60,000; 10,000 per class) of the FEVER training data. We consider two configurations: ProoFVer-K, which uses gold data to identify the transformed span for applying the predicted transformation, and ProoFVer-K-NoS, which instead only ensures that the predicted transformation occurs at least once in the final NatOp sequence.

System	Dev		Test	
	LA	Fever Score	LA	Fever Score
Using Retriever from Liu et al. (2020)				
ProoFVer	**80.23**	**78.17**	**79.25**	**74.37**
ProoFVer-MV	78.71	74.62	74.18	70.09
ProoFVer-A	79.83	76.33	77.16	72.47
ProoFVer-AR	77.42	75.27	–	–
KGAT	78.29	76.11	74.07	70.38
CorefBERT	79.12	77.46	75.96	72.30
Using Retriever from Stammbach (2021)				
ProoFVer-SB	**80.74**	**79.07**	**79.47**	**76.82**
DominikS	80.59	78.37	79.16	76.78

Table 3: Fact verification results on FEVER.

KEPLER		ProoFVer	
Training Data Size	Classifier Accuracy	-K-NoS (LA)	-K (LA)
1,800	69.07	64.65	66.73
6,000	74.02	68.86	72.41
18,000	79.67	74.25	76.23
30,000	80.61	75.39	77.76
45,000	82.76	77.62	78.84
60,000	84.85	78.61	79.67

Table 4: LA of ProoFVer-K and -NoS using predictions from KEPLER. Training data size used for KEPLER and its classifier accuracy is also provided.

6 Results

6.1 Fact Verification

Table 3 reports the fact verification results for ProoFVer and the baselines. Overall, ProoFVer-SB, our configuration using Stammbach's (2021) retriever, is the best performing model in our experiments. ProoFVer-SB, which outperforms Stammbach (2021) itself, is currently the highest scoring model in terms of label accuracy in the FEVER leaderboard. It also is the second best model in terms of FEVER Score, second only to the currently unpublished model titled ''mitchell.dehaven'', in the leaderboard.

ProoFVer, our default configuration using the retriever from Liu et al. (2020), differs from ProoFVer-SB only in terms of the retriever they use. ProoFVer is the best performing model among all the baselines and other ProoFVer configurations (-MV, -A, and -AR) that use Liu et al.'s (2020) retriever. As compared to ProoFVer-MV, ProoFVer's gains come primarily from its ability to handle multiple evidence sentences together, as opposed to handling each separately and then aggregating the predictions. 9.8% (1,960) of the claims in the FEVER development set require multiple evidence sentences for verification. While ProoFVer-MV predicts 60.1% of these instances correctly, ProoFVer correctly predicts 67.45% of these. Further, around 80.73% (of 18,038) of the single evidence instances are correctly predicted by ProoFVer-MV, in comparison to 81.62% instances for ProoFVer. Allowing the proof generator to infer the mutations dynamically, instead of having them predefined, benefits the overall performance of the model. The increasingly restricted variants with narrower search spaces (i.e., ProoFVer-A and ProoFVer-AR) lead to decreasing performances as shown in Table 3. ProoFVer-AR, the most restricted version, performs worse than all the other models.

Model	Dataset					
	FEVER-DEV			Symmetric FEVER		
	Original	FT	FT+L2	Original	FT	FT+L2
ProoFVer	**89.07±0.3**	**86.41±0.8**	**87.95±1.0***	**81.70±0.4**	**85.88±1.3**$^{\#}$	**83.37±1.3**$^{\#*}$
KGAT	86.02±0.2	76.67±0.3	79.93±0.9*	65.73±0.3	84.94±1.1$^{\#}$	73.34±1.5$^{\#*}$
CorefBERT	88.26±0.4	78.79±0.2	84.22±1.5*	68.49±0.6	85.45±0.2$^{\#}$	77.37±0.5$^{\#*}$

Table 5: Label accuracy of models on FEVER-development(DEV) and Symmetric FEVER with and without fine tuning. All results marked with $*$ and $\#$ are statistically significant (unpaired t-test) with $p < 0.05$ against their FT and Original variants respectively. FEVER-DEV predictions are using gold standard evidence.

Impact of Additional Manual Annotation Because the final filtering step in NatOp assignment (§4.2.2) requires additional manual annotation, we experimented with a proof set obtained without this step. Here, we arbitrarily select a NatOp sequence from the candidates remaining after the veracity label based filtering. The latter reduced the search space to just two possible NatOp sequences in 93.59% of the claims. However, training ProoFVer with these proofs resulted in a LA of 58.29% on the FEVER development set. In comparison, ProoFVer-K-NoS achieves a LA of 64.65%, even when using predictions from a KEPLER configuration trained on as little as 1,800 instances. Table 4 shows the LA for ProoFVer-K-NoS and ProoFVer-K when using KEPLER predictions, with varying training data sizes for KEPLER; the largest KEPLER configuration is trained on only 41.24% of claims in FEVER. Using this amount of training data, ProoFVer-K and ProoFVer-K-NoS achieve a LA of 79.67% and 78.61%, respectively. Here, ProoFVer-K outperforms all the baseline models, including CorefBert, which also uses additional annotation for pretraining.

6.2 Robustness

Symmetric FEVER As shown in Table 5, ProoFVer shows better robustness with a mean accuracy of 81.70% on the Symmetric FEVER test dataset, an improvement of 13.21% over CorefBERT, the next best model. All models improve their accuracy and are comparable on the test set when we fine-tune them on its development set. However, this results in more than 9% reduction on the original FEVER-DEV data for both the classifier based models, KGAT and CorefBERT. This catastrophic forgetting (French, 1999) occurs primarily due to the shift in label distribution during fine-tuning, as Symmetric FEVER contains only claims with Support and Refute labels. ProoFVer accuracy drops by only less than 3%, as it is trained with a seq2seq objective. To mitigate the effect of catastrophic forgetting, we apply L2 regularization (Thorne and Vlachos, 2021a), which improves all models on the FEVER development set. Nevertheless, ProoFVer has the highest accuracy on both FEVER and Symmetric FEVER among the competing models after regularization.

Generalizing to FEVER 2.0 ProoFVer when evaluated on FEVER 2.0 adversarial data, reports a LA of 82.79%, outperforming the previously best reported LA of 82.51% by Schuster et al. (2021). ProoFVer, after training on FEVER, is further fine-tuned (with L2 regularization) on heuristically generated proofs from the data contributed by the participants of the FEVER 2.0 shared task (disjoint from the evaluation set), and the proofs generated from the FEVER Symmetric data. On the other hand, Schuster et al. (2021) was trained on the VitaminC training data. When they further fine tune their default model with FEVER, their performance drops to 80.94%.

Stability Error Rate (SER): SER quantifies the rate of instances where a system alters its decision due additional evidence in the input, passed on by the retriever component. KGAT, CorefBERT, and DominikS have a SER of 12.35%, 10.27%, and 9.36% respectively. ProoFVer has an SER of only 6.21%, which is further reduced to 5.73% for ProoFVer-SB. The SER results confirm that the baselines change their predictions from Support or Refute after providing them with additional information more often than ProoFVer.

Claim: {No Country for Old Men}^{1} {was selected}^{2} {as the best of 2007}^{3} {by the grave digger}^{4}
Evidence: [No Country for Old Men]^1; The American Film Institute listed it as an AFI Movie of the Year, and the [National Board of Review]^4 [selected]^2 the film [as the best of 2007]^3.
NatOPs: ≡ ≡ ≡ ⇃↾ Label: REFUTES Human Rationale: National Board of Review

Claim: {Prague Castle}^{1} {feeds}^{2} {over 1.8 million}^{3} {visitors annually}^{4}
Evidence: [Prague Castle]^1; The .... in Prague [attracting]^2 [over 1.8 million]^3 [visitors annually]^4.
NatOPs: ≡ # ≡ ≡ Label: NOT ENOUGH INFO Human Rationale: attracts

Claim: {Southampton F.C.}^{1} {is a soccer team}^{2}
Evidence: [Southampton F.C.]^1; Southampton ... [is a professional association football club]^2 based ... football.
NatOPs: ≡ ⊑ Label: SUPPORTS Human Rationale: football club

Figure 6: Human rationale extraction for predicted proofs from ProoFVer. The claim and evidence spans are enclosed within '{ }' and '[ ]', respectively, with numbered superscripts showing the correspondence between the spans. The predicted rationales are underlined and the portions matching with the human rationales are highlighted.

6.3 ProoFVer Proofs as Explanations

6.3.1 Rationale Extraction

Rationales extracted based on attention are often used as means to highlight the reasoning involved in the decision making process of various models (DeYoung et al., 2020). For this evaluation, we compare using token-level F-score of the predicted rationales with human-provided rationales for 300 claims from the FEVER development data, as elaborated in Section 5.1. We ensure that all the systems are provided with the same set of evidence sentences, and consider only those words from the evidence as rationales that do not occur in the claim. For ProoFVer, we additionally remove evidence spans which are part of mutations with an equivalence NatOp. For KGAT and CorefBERT, we obtain the rationales by sorting the eligible words in descending order of their attention scores, and for each instance we find the set of words with the highest token overlap F-score with the rationale. Here, we consider the words in the top 1% of attention scores, and also those ranging from 5% to 50% of the words in step sizes of 5%. We find that ProoFVer achieves a token level F-score of 93.28, compared to 87.61 and 86.42, the best F-Scores for CorefBERT and KGAT. Figure 6 shows the rationales for 3 instances extracted from ProoFVer, one for each label. All the three proofs result in correct decisions. While for the first two claims there is a perfect overlap with the human rationale, the third claim in Figure 6 has some extraneous information in the predicted proof.

6.3.2 Human Evaluation

≡	Equivalent Spans
↿⇂	Evidence span contradicts the claim span
⊑	Claim span follows from evidence span
⊑	(Insert) New information from evidence
⊒	Incomplete Evidence
⋏	Evidence span refutes claim span
⋏	Claim span negated (Deletion)
#	Unrelated claim span and evidence span
#	No related evidence found (Deletion)

Table 6: NatOPs and the corresponding paraphrases.

We use forward prediction (Doshi-Velez and Kim, 2017) here, where humans are asked to predict the system output based on the explanations. For assessing ProoFVer, we provide the claim, the proof as the explanation, and those evidence sentences from which the evidence spans in the proof were extracted. Since we are interested in evaluating the applicability of our proofs as natural language explanations, we ensure that none of our subjects are aware of the deterministic nature of determining the label from natural logic proofs. Moreover, we replaced the NatOps in the proof with plain English phrases for better comprehension by the subjects, as shown in Table 6. As the baseline setup for comparison, we provide the claim with all five retrieved evidence sentences.

We form a set of 24 different claims, 12 each from ProoFVer and baseline, and 3 individual subjects independently annotate the same set. Finally, we altogether obtain annotations for 5 sets, resulting in 60 claims, 120 explanations, and a total of 360 annotations from 15 subjects.[2] For all 60

[2] Although 19 subjects volunteered, one of them annotated a set that did not receive any other annotations. In another set, two of them had prior knowledge in natural logic, leading to disqualification of these 3 annotations from the set.

i) Claim: {Psych}^{1} {is **<u>neither</u>** a drama}^{2} {**<u>nor</u>** comedy}^{3}
Evidence: [Psych]^1 is an American detective [comedy]^3 [drama]^2 television series ... on ION Television .
NatOPs: ≡⋏⋏ Predicted Label: Supports Groundtruth Label: Refutes

ii) Claim: {Hot Right Now}^{1} {is **<u>mistakenly</u>** attributed}^{2} {to DJ Fresh}^{3}
Evidence: '‘[Hot Right Now]^1'' [is a single by]^2 British ... producer [DJ Fresh]^3, released ... Nextlevelism.
NatOPs: ≡⊑≡ Predicted Label: Supports Groundtruth Label: Refutes

iii) Claim: {N. Vietnam}^{1} {existed **<u>from 1934</u>**}^{2} {**<u>to 1940</u>**}^{3} {and at no other time}^{4}
Evidence: [N. Vietnam]^1, ... , was a state in Southeast Asia which existed from [1945]^2 to [1976]^3. [DEL]^4
NatOPs: ≡⥯⥯⊒ Predicted Label: Not Enough Info Groundtruth Label: Refutes

Figure 7: Cases of incorrect proof generation from ProoFVer. The claim and evidence spans are enclosed within '{ }' and '[ ]', respectively, with numbered superscripts showing the correspondence between the spans.

claims, ProoFVer, CorefBERT, and KGAT predicted the same labels, though not necessarily the correct ones (the subjects were not aware of this). All the subjects were pursuing a PhD or postdocs in fields related to computer science and computational linguistics, or industry researchers/data scientists.

With ProoFVer's proofs, subjects are able to predict the model decisions correctly in 81.67% of the cases as against 69.44% of the cases with only the evidence. In both setups, subjects were often confused on instances with a Not Enough Info label, and the forward predictions were comparable, with 66.67% (ProoFVer) and 65% (baseline). In many such cases, subjects subconsciously filled in their own world knowledge that is not found in the evidence to arrive at a Support or Refute decision. Further, for instances with both Refute and Support labels, subjects correctly predicted ProoFVer's decisions 86.67% and 91.67% times, respectively, against only 70% and 73.33% for the baseline. The inter-annotator agreement for ProoFVer's explanations is 0.7074 in Fleiss κ (Fleiss, 1971), and 0.6612 for the baseline.

7 Limitations

Figure 7 shows three instances of incorrect proofs from ProoFVer, which highlight some of the well known limitations in natural logic (Karttunen, 2015; MacCartney, 2009). In Figure 7.i, the claim uses two negation words, ''neither'' and ''nor'', both of which appear in different spans and lead to prediction of two negation NatOps. However, this NatOp sequence nullifies the effect of the negation NatOp and predicts Support instead of Refute. Similarly, in Figure 7.ii the adverb ''mistakenly'' negates semantics of the verb. However, its effect is not captured in the second mutation and ProoFVer predicts the forward entailment NatOP, leading to the Support label. Moreover, the NatOP sequence remains the same even if we remove the term ''mistakenly'' from the claim, demonstrating that the effect of the adverb is not captured by our model. Similar challenges involving adverbs and non-subsective adjectives (Pavlick and Callison-Burch, 2016) when performing inference in natural logic have been reported in prior work (Angeli and Manning, 2014).

In Figure 7.iii, the claim states a time period by mentioning its start and end years, which appear in two different claim spans. However, ProoFVer does not capture the sense of the range implied by the spans containing ''from 1934'' and ''to 1940''. Instead, two similar 4-digit number patterns are extracted from the evidence and are directly compared to the claim spans, resulting in two alternation NatOps, thereby predicting Not Enough Info. Handling such range expressions is beyond the expressive power of the natural logic, and often other logical forms are needed to perform such computations (Liang et al., 2013). Datasets like FEVEROUS (Aly et al., 2021), which consider semi-structured information present in tables, often require such explicit computations for which approaches purely based on natural logic are not sufficient.

Finally, ProoFVer, due to its auto-regressive formulation, generates the corresponding evidence spans and NatOps for the claim spans sequentially from left to right. However, the steps in the natural logic based inference are not subject to any such specific ordering, and hence the order in which the NatOPs are generated is non deterministic by default (Angeli and Manning, 2014). ProoFVer benefits from the implicit knowledge encoded in the pretrained language models, specifically BART, which follows auto-regressive decoding. Nevertheless, in the future we plan to experiment with alternative decoding approaches, including some of the recent developments in non-autoregressive conditional language models (Xu and Carpuat, 2021) and transformer-based proof generators (Saha et al., 2021).

8 Conclusion

We presented ProoFVer, a natural logic-based proof system for fact verification. Currently, we report the best results in terms of label accuracy, and the second best results in FEVER Score in the FEVER leaderboard. Moreover, ProoFVer is more robust in handling superfluous information from the retriever, and handling counterfactual instances. Finally, ProoFVer 's proofs are faithful explanations by construction, and improve the understanding of the decision making process of the models by humans.

Acknowledgments

Amrith Krishna and Andreas Vlachos are supported by a Sponsored Research Agreement between Facebook (now Meta) and the University of Cambridge. Andreas Vlachos is additionally supported by the ERC grant AVeriTeC (GA 865958) and the EU H2020 grant MONITIO (GA 965576). The authors would like to thank Nicola De Cao for helpful conversations, and Dominik Stammbach for sharing the retriever data. Further, the authors would like to thank all the 19 subjects who volunteered to be part of the human evaluation, namely, James Thorne, Michael Sejr Schlichtkrull, Ashim Gupta, Bishal Santra, Ashutosh Soni, Abhik Jana, Sudipa Mandal, Madhumita Mallick, Roshni Tayal, Sandipan Sikdar, Rishabh Kumar, Jasabanta Patro, Unni Krishnan, Anirban Santara, Jivnesh Sandhan, Soumya Sarkar, Devaraj Adiga, Aniruddha Roy, and Muskan Garg. Finally, we would like to thank the anonymous reviewers, the action editor, and the editor in chief Brian Roark for their valuable feedback.

References

Lasha Abzianidze. 2017a. LangPro: Natural language theorem prover. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 115–120, Copenhagen, Denmark. Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-2020

Lasha Abzianidze. 2017b. *A Natural Proof System for Natural Language*. Ph.D. thesis, Tilburg University.

Naser Ahmadi, Joohyung Lee, Paolo Papotti, and Mohammed Saeed. 2019. Explainable fact checking with probabilistic answer set programming. In *Proceedings of the 2019 Truth and Trust Online Conference (TTO 2019), London, UK, October 4–5, 2019*. https://doi.org/10.36370/tto.2019.15

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. 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. Association for Computational Linguistics.

Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. FEVEROUS: Fact Extraction and VERification Over Unstructured and Structured information. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Gabor Angeli. 2016. *Learning Open Domain Knowledge From Text*. Ph.D. thesis, Stanford University.

Gabor Angeli and Christopher D. Manning. 2014. NaturalLI: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 534–545, Doha, Qatar. Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1059

Gabor Angeli, Neha Nayak, and Christopher D. Manning. 2016. Combining natural logic and shallow reasoning for question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 442–452, Berlin, Germany. Association for Computational Linguistics. https://doi.org/10.18653/v1/P16-1042

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over Wikipedia graph for question answering. In *International Conference on Learning Representations*.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. Generating fact checking explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7352–7364, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.656

Roy Bar-Haim, Ido Dagan, Iddo Greental, Idan Szpektor, and Moshe Friedman. 2007. Semantic inference at the lexical-syntactic level for textual entailment recognition. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 131–136, Prague. Association for Computational Linguistics. https://doi.org/10.3115/1654536.1654563

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.408

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. ArXiv Preprint arXiv:1702.08608v2.

Angela Fan, Aleksandra Piktus, Fabio Petroni, Guillaume Wenzek, Marzieh Saeidi, Andreas Vlachos, Antoine Bordes, and Sebastian Riedel. 2020. Generating fact checking briefs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7147–7161, Online. Association for Computational Linguistics.

Yufei Feng, Zi'ou Zheng, Quan Liu, Michael Greenspan, and Xiaodan Zhu. 2020. Exploring end-to-end differentiable natural logic modeling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1172–1185, Barcelona, Spain (Online). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.101

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382. https://doi.org/10.1037/h0031619

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135. https://doi.org/10.1016/S1364-6613(99)01294-2

Mohamed H. Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. 2019. Exfakt: A framework for explaining facts over knowledge graphs and text. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pages 87–95, New York, NY, USA. Association for Computing Machinery. https://doi.org/10.1145/3289600.3290996

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online. Association for Computational Linguistics.

Andreas Hanselowski, Christian Stab, Claudia Schulz, Zile Li, and Iryna Gurevych. 2019. A richly annotated corpus for different tasks in automated fact-checking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 493–503, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/K19-1046

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5516

Stefan Harmeling. 2009. Inferring textual entailment with a probabilistically sound calculus. *Natural Language Engineering*, 15(4):459–477. https://doi.org/10.1017/S1351324909990118

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced bert with disentangled attention.

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019, Los Angeles, California. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. https://doi.org/10.5281/zenodo.1212303

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. Association for Computing Machinery. https://doi.org/10.1145/1014052.1014073

Thomas F. Icard III and Lawrence S. Moss. 2014. Recent progress on monotonicity. In *Linguistic Issues in Language Technology, Volume 9, 2014 - Perspectives on Semantic Representations for Textual Inference*. CSLI Publications. https://doi.org/10.33011/lilt.v9i.1325

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.386

Alon Jacovi and Yoav Goldberg. 2021. Aligning Faithful Interpretations with their Social Attribution. *Transactions of the Association for Computational Linguistics*, 9:294–310. https://doi.org/10.1162/tacl_a_00367

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C. Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.409

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1627–1643, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.147

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1215

Isaac Johnson. 2020. Analyzing wikidata transclusion on English Wikipedia. In *Proceedings of the 1st Wikidata Workshop (Wikidata 2020) co-located with 19th International Semantic Web Conference(OPub 2020), Virtual Conference, November 2–6, 2020*, volume 2773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Lauri Karttunen. 2015. From natural logic to natural reasoning. In *Computational Linguistics and Intelligent Text Processing*, pages 295–309, Cham. Springer International Publishing. https://doi.org/10.1007/978-3-319-18111-0_23

Neema Kotonya and Francesca Toni. 2020a. Explainable automated fact-checking: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5430–5443, Barcelona, Spain (Online). International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.474

Neema Kotonya and Francesca Toni. 2020b. Explainable automated fact-checking for public health claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7740–7754, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.623

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics. https://doi.org/10.18653/v1/D16-1011

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. 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. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.703

Percy Liang, Michael I. Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446. https://doi.org/10.1162/COLI_a_00127

Zachary C. Lipton. 2018. The mythos of model interpretability. *Communications of ACM*, 61(10):36–43. https://doi.org/10.1145/3233231

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Bill MacCartney. 2009. *Natural Language Inference*. Ph.D. thesis, Stanford University.

Bill MacCartney and Christopher D. Manning. 2007. Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200, Prague. Association for Computational Linguistics. https://doi.org/10.3115/1654536.1654575

Yashar Mehdad. 2009. Automatic cost estimation for tree edit distance using particle swarm optimization. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 289–292, Suntec, Singapore. Association for Computational Linguistics. https://doi.org/10.3115/1667583.1667672

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41. https://doi.org/10.1145/219717.219748

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2553–2566, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1258

Ellie Pavlick and Chris Callison-Burch. 2016. So-called non-subsective adjectives. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 114–119, Berlin, Germany. Association for Computational Linguistics. https://doi.org/10.18653/v1/S16-2014

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics. https://doi.org/10.3115/v1/P15-2070

Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2020. Revisiting evaluation of knowledge base completion models. In *Automated Knowledge Base Construction.*

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking fake news and false claims using evidence-aware deep learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 22–32, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1003

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. 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 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics. https://doi.org/10.18653/v1/N18-1119

Swarnadeep Saha, Prateek Yadav, and Mohit Bansal. 2021. multiPRover: Generating multiple proofs for improved interpretability in rule reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3662–3677, Online. Association for Computational Linguistics.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! Robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.52

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1341

Dominik Stammbach. 2021. Evidence selection as a token-level prediction task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 14–20, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.fever-1.2

Asher Stern, Roni Stern, Ido Dagan, and Ariel Felner. 2012. Efficient search for transformation-based inference. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 283–291, Jeju Island, Korea. Association for Computational Linguistics.

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 41–48, Barcelona, Spain. Association for Computational Linguistics.

James Thorne and Andreas Vlachos. 2021a. Elastic weight consolidation for better bias inoculation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 957–964, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.eacl-main.82

James Thorne and Andreas Vlachos. 2021b. Evidence-based factual error correction. 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 3298–3309, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-long.256

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018a. FEVER: A large-scale dataset for fact extraction and VERification. 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 809–819, New Orleans, Louisiana. Association for Computational Linguistics. https://doi.org/10.18653/v1/N18-1074

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018b. The fact extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5501

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2019. The FEVER2.0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 1–6, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-6601

Joseph E. Uscinski and Ryden W. Butler. 2013. The epistemology of fact checking. *Critical Review*, 25(2):162–180. https://doi.org/10.1080/08913811.2013.843872

V. M. S. Valencia. 1991. *Studies on Natural Logic and Categorial Grammar*. Ph.D. thesis, University of Amsterdam.

Johan Van Benthem. 1986. Natural logic. In *Essays in Logical Semantics*, pages 109–119, Dordrecht. Springer Netherlands. https://doi.org/10.1007/978-94-009-4540-1_6

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85. https://doi.org/10.1145/2629489

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.609

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194. https://doi.org/10.1162/tacl_a_00360

Lianwei Wu, Yuan Rao, Yongqiang Zhao, Hao Liang, and Ambreen Nazir. 2020. DTCA: Decision tree-based co-attention networks for explainable claim verification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1024–1035, Online. Association for Computational Linguistics.

Weijia Xu and Marine Carpuat. 2021. EDITOR: An edit-based transformer with repositioning for neural machine translation with soft lexical constraints. *Transactions of the Association for Computational Linguistics*, 9:311–328.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential reasoning learning for language representation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7170–7186, Online. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, UAI'05, pages 658–666, Arlington, Virginia, USA. AUAI Press.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.549