2021 IEEE Third International Conference on Cognitive Machine Intelligence (CogMI) | 978-1-6654-1621-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/COGMI52975.2021.00015

# TRIPLEX: Triple Extraction for Explanation

Mattia Setzu
*Dept. of Computer Science*
*University of Pisa*
*Pisa, Italy*
*mattia.setzu@phd.unipi.it*

Anna Monreale
*Dept. of Computer Science*
*University of Pisa*
*Pisa, Italy*
*anna.monreale@phd.unipi.it*

Pasquale Minervini
*Dept. of Computer Science*
*University College London*
*London, United Kingdom*
*p.minervini@ucl.ac.uk*

***Abstract*—Transformer-based models are used to solve a variety of Natural Language Processing tasks. Still, these models are opaque and poorly understandable for their users. Current approaches to explainability focus on token importance, in which the explanation consists of a set of tokens relevant to the prediction, and natural language explanations, in which the explanation is a generated piece of text. The latter are usually learned by design with models trained end-to-end to provide a prediction and an explanation, or rely on powerful external text generators to do the heavy lifting for them. In this paper we present TRIPLEX, an explainability algorithm for Transformer-based models fine-tuned on Natural Language Inference, Semantic Text Similarity, or Text Classification tasks. TRIPLEX explains Transformers-based models by extracting a set of facts from the input data, subsuming it by abstraction, and generating a set of weighted triples as explanation.**

***Keywords*-Explainable Artificial Intelligence; Transformer-based models; Natural Language Inference**

## I. INTRODUCTION

Attention-based models, such as Transformer-based models [1], have become the de-facto standard in many Natural Language Processing (NLP) tasks. Transformer-based models such as BERT [2], RoBERTa [3], and DeBERTa [4] produce very accurate results in a variety of NLP tasks [5], [6], including Question Answering, Natural Language Inference, Sentiment Classification, and Question Answering. Pre-trained as Masked Language Models (MLM) [2], Transformers can then be fine-tuned on a variety of tasks, granting them a high degree of flexibility. With a broad spectrum of complex tasks to perform, Transformers tend to be large models with millions [2], if not billions [4], of parameters, effectively making them black boxes to anyone who tries to understand their predictions.

Here, we focus on Natural Language Inference (NLI) tasks [7], Semantic Text Similarity (STS) and Text Classification (TC). Given two sentences, namely *premise* and *hypothesis*, an NLI task consists in identifying the relationship between them: *entailment* if the premise entails the hypothesis, *contradiction* if the premise contradicts the hypothesis, and *neutrality* if the premise is inconsequential to the hypothesis. To illustrate, consider the following example:

**Premise** Mice given a substance found in red wine lived longer despite a fatty diet, a study shows.

**Hypothesis** Mice fed with red wine lived longer despite a fatty diet.

The two are in an entailment relation, as the hypothesis follows from the premise. Different hypotheses can yield to different results; for example,

**Hypothesis** Mice fed with red wine *did not live as long* despite a fatty diet.

yields a *contradiction* label, while

**Hypothesis** Mice fed with *orange juice* lived longer despite a fatty diet.

yields a *neutrality* label.

STS and TC are more general tasks in which we rank the similarity of a pair of text excerpts and classify a given text excerpt, respectively.

In this work, we argue that large-scale automated decision systems and the neural models at their core ought to be thoroughly understood by the means of *explanations*. Explainable Artificial Intelligence (XAI) offers algorithmic solutions falling in one of two categories, either token importance (TI) or natural language (NL) explanations. Token importance explanations provide a set of relevant tokens in the input text, possibly with an associated importance score, of the relevance each token had on the prediction of the model. This family of explanations heavily relies on the input permanence assumption, i.e., they assume that even at explanation time the user has access to, and should fully leverage the input text.

Tokens assume a semantically valid meaning only when read *in situ* in the input text, hence as explanations they are *incomplete*, as they are not meaningful *on their own* without the support of the input text. Natural language explanations instead rely on learning appropriate natural language excerpts, thus providing rationales for a given prediction. Most explanation algorithms in this category operate either on a by-design or generative approach. In the former, the explanation model and the black-box model coincide, and an explanation is generated by-design alongside a prediction [8]. In the latter, an external natural language generator is leveraged to generate an explanation, and the model is queried to guide the explanation generation and verify that the explanation is consistent with the model prediction [9]. These algorithms

978-1-6654-1621-4/21/$31.00 ©2021 IEEE
DOI 10.1109/CogMI52975.2021.00015

tend to provide reasonably good explanations, but rarely rely on semantically meaningful explanation generation while fully relying on the goodness of the generator. This makes for methods that are able to explain only as much as the generator is able to generate, and lack an inherent semantic component that we may find, for instance, in structured knowledge bases. We argue that while faithful to the model, leveraging structured domain knowledge makes for more commonsense explanations that instead of relying on the knowledge of a language generator, rely on human domain knowledge. Moreover, we argue for minimizing reliance on external language generator models in explainability, as we prefer to avoid to rely on a black-box model to explain another.

In this work, we aim to generate *self-contained* explanations, i.e., explanations comprehensible without the support of the input text, for Transformer-based models trained on NLI, STS, or TC tasks. We wish to generate explanations acting as surrogates for the instance at hand, and to completely remove language generators from our explanation pipeline, replacing them with a combination of information extraction algorithms and semantic perturbations. We argue that a good surrogate should be explainable, while still being coherent with the decision at hand. Our explanations aim to emulate two natural reasoning mechanisms, one *by abstraction*, and one *by similarity*, according to the task at hand. In particular, we rely on abstraction for explaining NLI tasks, and on similarity for STS and TC tasks. To this end, we leverage Information Extraction algorithms and knowledge bases explicitly designed to encode *supertype* and *same type* relationship between concepts. The former extract a set of relevant facts from the input text, the latter create surrogate triples to provide as explanations. We name our proposed algorithm TRIPLEX (Triples for Explainability).

The rest of the paper is as follows. In Section II we give a brief background on the literature, in Section III we present TRIPLEX, in Section IV we show our experiments, and in Section V we conclude our paper.

## II. BACKGROUND AND RELATED WORK

Before introducing TRIPLEX, we discuss the literature and introduce some basic notions to understand the details of our proposal.

*Transformers:* Transformers employ a multi-head self-attention mechanism in which attention matrices are computed on a set of learned token representations. Indicated as *keys* $K$ and *values* $Q$, attention on tokens is computes as follows:

$$\text{Att}_h(K, Q, V) = \text{softmax}\left(\frac{QK^T}{d}\right) V_h, \qquad (1)$$

where $d$ is a normalization parameter, and $V_h$ is a set of learned parameters mapping the keys and values to different representations. Other than the scaled dot-product [1] shown in Equation (1), attention is estimated in many forms, from cosine similarity [10], to dot-product [11]. Transformers learn a set of representation transformations $V_1, \ldots, V_h$, each yielding a different attention *head*, as to have an ensemble of token representations, with the goal of having different heads learning different attentions. Heads are then arranged in sequential *attention layers*, some specializing in different tasks [12], [13], [14].

While not an explanation per se [15], attention weights encode an alignment between tokens. In our proposal, we leverage attention heads as an alignment indicator between explanations and hypotheses.

*Explainability in NLP Models:* Token importance explanations provide the user with a set of relevant input tokens, effectively "highlighting" the input sequence most relevant to the prediction. They come both post-hoc, as is the case for Shap [16], LIME [17], and Integrated Gradients [18], or by-design, as is the case for Rationales [19], and other models [20], [21], [22], [23]. Post-hoc algorithms provide explanations *after* the model prediction while by-design algorithms do so out of the box. By-design explanation algorithms usually rely on two-stage neural pipeline comprised of an explanation generator, tasked with generating an explanation, and a predictor, tasked with performing the learning task with the explanation as input. Some exclusively rely on the generated explanation [19] while others rely both on the generated explanation and input [23]. To accommodate the interest in token importance algorithms, some datasets go as far as providing token highlights as input [8], allowing to train models on gold truth explanations. Similar and relevant tasks involve combinations of deductive [24], [25], abductive [26] and commonsense [27], [28], [29], [30] reasoning, in which models are trained to yield a possibly structured reasoning to go along the prediction.

Token importance has been the main focus of XAI on natural language models, recently the focus has shifted towards natural language explanations, in which the explanation is a realistic, synthetic piece of text. We find applications in Question Answering [31] and Text Classification [32]. On NLI tasks, the algorithm presented in [33] and NILE [9] are of particular interest. Silva et al. [33] propose a by-design explanation algorithm that enriches premise and hypothesis with additional knowledge from a knowledge base to estimate the entailment label. The notions relevant for the prediction found in the knowledge base is then used to generate a natural language explanation. NILE is a post-hoc explanation algorithm that leverages a two-way architecture by generating candidate natural language explanations for each label, and delegating prediction to a second module that selects a candidate and its associated label.

*Knowledge Bases and Semantic Perturbations:* Knowledge bases offer a formal and machine-readable approach to domain knowledge representation. They range from general purpose knowledge bases such as DBpedia [34] and

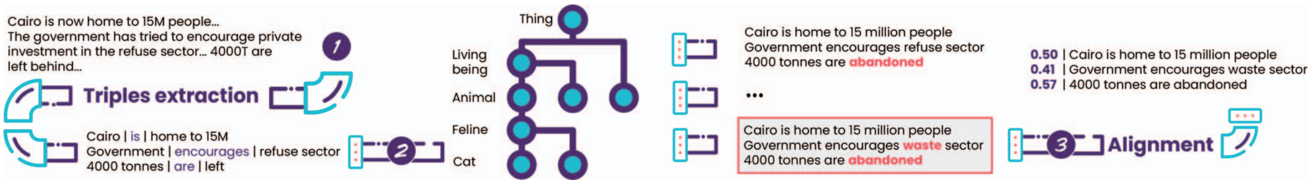



Figure 1: An example of the TRIPLEX pipeline on a NLI task. In stage *1* we extract a set of triples from the premise/input text to use as an explanation stub. The stub is forwarded to stage *2*, where a knowledge-aware module generalizes it into a set of explanation candidates by leveraging the WordNet taxonomy. The most generalized among the coherent candidates is fed to the stage *3* which that computes the *alignment* of each triple, yielding the final explanation.

YAGO [35], to domain-specific ones such as Bio2RDF [36] and Hetionet [37] for life sciences and WordNet [38] for linguistics, and application-driven graphs such as the Google Knowledge Graph, Microsoft's Bing Knowledge Graph, and Facebook's Social Graph [39]. Recently, commonsense knowledge bases have started to gain traction in the community [40], [27], [41]. WordNet provides taxonomies of concepts with *synonym* and *hypernym* relations, i.e., each concept $c$ is associated to a set of synonyms $\text{syn}(c)$ and abstractions and $\text{hyper}(c)$. For instance, the concept of `cat` is associated with hypernyms `feline` and `quadruped`, which are then associated with hypernyms `mammal` and `animal`, and so on and so forth. `cat` is also associated to synonyms `kitty` and `kitten`. Hypernyms provide direct and simple semantic relationships to abstract any concept in a given text, thus they are suitable for our "reasoning by abstraction" purposes, while synonyms are naturally suitable for "reasoning by similarity".

*Conditioned Text Generation:* Conditioned Text Generation (CTG) is a Sequence-to-Sequence task in which a given text $t$ is transformed in another $t'$ conditioned on some *control code* $c$. Control codes allow to automatically add, remove [32], replace [42] some information from the text, to change its sentiment or the context of the discussed topic [43], and even generate novel text from a given prompt according to the topic embedded in the control code [44].

*Information Extraction:* Information Extraction algorithms [45], [46], [47], [48] extract factual information from a given text, providing a structured representation in the form of `subject-predicate-object` triples, each indicating a relationship of type `predicate` between the entities `subject` and `object`. For instance, let us take the example in the introduction, `Mice fed with red wine lived longer despite a fatty diet.`, from which [45], which extracts `(Mice, fed with, wine)` and `(Mice, fed with, red wine)`.[1]

## III. TRIPLEX

TRIPLEX locally explains predictions of Transformer-based models fine-tuned on NLI, TC, or STS tasks via natural language explanations in the form of triples. TRIPLEX consists of a three-stage pipeline, as shown in Figure 1: i) an *information extraction* module $I$, tasked with extracting natural language triples $e$, operating as the explanation stub; ii) a *knowledge base* $G$, tasked with guiding the semantic perturbation of said triples; and iii) an *explanation enrichment* $E$, tasked with enriching the explanation $E(e)$ for the user.

Our proposal is straightforward: the information extraction module extracts the explanation stub $e$ from the input, which is then semantically perturbed in accordance with the knowledge base $G$, generating a set of candidate explanations $\widetilde{E}$, each generalizing $e$. Finally, the explanation extractor selects an explanation in $\widetilde{E}$ to provide to the user and enriches it with an optional *alignment score* enabling a rank of the triples according to their importance.

We detail each phase separately, and report the full procedure in Algorithm 2.

*Information Extraction:* In a first step of Information Extraction we extract a set of triple-like propositions $e$ from the input text/premise by leveraging OpenIE [45], OllIE [46], and ClausIE [48]. In other words, this step yields a baseline explanation stub $e$ that we are then going to perturb. For NLI tasks, we limit ourselves exclusively to extraction from and perturbation of the premise for generating explanations that provide the most general conditions under which the same premise-hypothesis relationship is maintained. For STS and TC tasks instead we rely on the whole input text.

*Candidate explanations generation and selection:* In this second step, we perturb the explanation stub through the use of a knowledge base. In particular, the knowledge base enables perturbations of the explanation stub either by generalization (NLI tasks) or similarity (STS, TC tasks). We leverage this process to generate a set of candidate explanations stemming from the stub. Finally, we select the best candidate explanation that we forward to the final stage.

As knowledge base $G$ we employ WordNet [49], and leverage its hypernym and synonyms taxonomy to perturb the given text, iteratively constructing more general/different text. Hypernym chains allow us to create abstractions of concepts, thus enabling more abstract reasoning. Given a concept $c$ in $G$, we indicate with $\text{hyper}(c, \gamma)$ its immediate

[1] Available at https://github.com/philipperemy/stanford-openie-python

**Algorithm 1** Explansion algorithm: generates a set of terms, either hypernyms or synonyms) from the given input $x$ according to the required strategy $S$.

**Input:** Input $x$, maximum depth $K$, search radius $\gamma$, knowledge base $G$, strategy $S$
**Output:** Expanded terms $T$

**function** EXPAND($x$, $K$, $\gamma$, $G$, $S$)
    **if** $S ==$ 'hypernyms' **then**
        $x' \leftarrow x[0]$ ▷ select only the premise
        $T \leftarrow$ HYPERNYMS$(x', K, \gamma, G)$ ▷ create hypernyms
    **else**
        $T \leftarrow$ SYSNONYMS$(x, K, \gamma, G)$ ▷ create synonyms
    **return** $T$

**Algorithm 2** Explanation algorithm: given a text excerpt $t$, TRIPLEX creates an explanation stub by extracting a set of triples by Information Extraction (Line 4). The stub is then provided to a knowledge base-powered algorithm that generates a set of explanation candidates (Line 5) by semantically perturbing it according to the knowledge base. We query again the model to classify candidates in terms of label (Line 10), and to compute the hypernym/synonym distance to the baseline explanation (Line 11). Finally, we compute the alignment score of each triple with respect to the hypothesis(Line 18)/input text (Line 21), and return it alongside the explanation (Line 22).

**Input:** Input $x \in \{(p_i, h_i), t_i\}$, Information extractor $I$, Transformer-based model $f$, knowledge graph $G$, maximum depth $K$, search radius $\gamma$, head $h$, layer $l$, Strategy $S$
**Output:** explanation $e$, alignment score $a$

1: **function** TRIPLEX($t$, $f$, $K$, $\gamma$, $G$, $S$)
2:     $y \leftarrow f(x)$
3:     $\widetilde{E} \leftarrow \emptyset$
4:     $e \leftarrow \mathrm{I}(t, S)$ ▷ create explanation stub
5:     $H \leftarrow$ EXPAND$(e, K, \gamma, G, S)$ ▷ create explanation stub according to the task
6:     $\widetilde{E} \leftarrow$ PERTURB$(e)$ ▷ perturb
7:     candidates $\leftarrow []$
8:     distances $\leftarrow []$
9:     **for** $\widetilde{e} \in \widetilde{E}$ **do**
10:         $\widetilde{y}_i \leftarrow f(\widetilde{e})$
11:         $d_{\widetilde{e}} \leftarrow$ distance$(G, e, \widetilde{e})$ ▷ compute candidate distance
12:         **if** $\widetilde{y}_i = y$ **then** ▷ select explanations
13:             candidates $\leftarrow$ APPEND(candidates, $\widetilde{e}$) ▷ add explanation
14:             distances $\leftarrow$ APPEND(distances, $d_{\widetilde{e}}$) ▷ add explanation distance
15:     $e^* \leftarrow$ candidates$[\arg\max_t$ distances$[t]]$ ▷ select explanation
16:     **if** $S ==$ 'hypernyms' **then**
17:         **for** $e_t \in e^*$ **do**
18:             $a_t \leftarrow \alpha_i^{h,l}(e_t, x_i[1])$ ▷ alignment to hypothesis
19:     **else**
20:         **for** $e_t \in e^*$ **do**
21:             $a_t \leftarrow \alpha_i^{h,l}(e_t, x)$ ▷ alignment to whole input
22:     **return** $e^*, a$

top-$\gamma$ hypernyms, that is the top-$\gamma$ hypernyms one level higher in the taxonomy. Similarly, we indicate with $\mathrm{syn}(c, \gamma)$ its immediate top-$\gamma$ synonyms, that is the top-$\gamma$ synonyms in the taxonomy. For simplicity, we are going to illustrate the algorithm focusing on $\mathrm{hyper}$, the procedure followed when using $\mathrm{syn}$ is analogous. We repeat this procedure on each hypernym (synonym), up to $K$ level up in the taxonomy. At each application of $\mathrm{hyper}$ we again select the top-$\gamma$ hypernyms according to their likelihood. Each successive application of $\mathrm{hyper}$ yields hypernyms at different levels of abstractions, each increasing the distance in the taxonomy between the initial concept $c$ and the yielded hypernym. In our previous `cat` example, $\mathrm{hyper}(\texttt{cat})$ yields $\{\texttt{feline}, \texttt{quadruped}\}$, a set of hypernyms at distance 1.

**Premise**

Cairo is now home to some 15 million people – a burgeoning population that produces approximately 10,000 tonnes of rubbish per day, putting an enormous strain on public services. In the past 10 years, the government has tried hard to encourage private investment in the refuse sector, but some estimate 4,000 tonnes of waste is left behind every day, festering in the heat as it waits for someone to clear it up. It is often the people in the poorest neighborhoods that are worst affected. But in some areas they are fighting back. In Shubra, one of the northern districts of the city, the residents have taken to the streets armed with dustpans and brushes to clean up public areas which have been used as public dumps.

**Hypothesis**

15 million tonnes of rubbish are produced daily in Cairo.

**Explanation**

| Alignment | Rank | Subject | Predicate | Object |
|---|---|---|---|---|
| .050 | 2 | Cairo | is | home to some 15 million people |
| .041 | 5 | Government | encourage | finance in waste sector |
| .043 | 4 | Finance | is | in waste sector |
| .046 | 3 | People | are | in poor neighborhood |
| .057 | 1 | 4000 tonnes | are | left |

Table I: Example of a TRIPLEX explanation for a *contradiction* instance. Triples in the (subject, predicate, object) form are shown with their alignment score (column **Alignment**) and with their rank order (column **Rank**). Extraction spans are shown in blue in the original premise.

Applying hyper again to `feline` and `quadruped` will then yield `mammal` and `living being`, both hypernyms at distance 2. Formally, the *hypernym distance* $d_h$ between two concepts $c, c'$ in $G$ is the number of levels separating them in $G$. When dealing with multiple pairs of concepts $C = \{(c_1, c'_1), \dots, (c_n, c'_n)\}$ their hypernym distance is defined as the sum of their hypernym distances, i.e.:

$$d_H(C) = \sum_{(c_i, c'_i) \in C} d_h(c_i, c'_i). \tag{2}$$

The larger the distance, the more abstract one set of concepts with respect to the other. In TRIPLEX, we leverage the hypernym distance to select the most abstract candidate explanation. The same goes analogously for synonyms.

*Explanation enrichment:* Let $D = (X, Y)$ be a dataset of $n$ instances $\{x_i\}_{i=1}^n$ and labels $\{y_i\}_{i=1}^n$. In NLI tasks, we denote with $X = \{x_i = (p_i, h_i)\}_{i=1}^n$ the set of input premises and hypotheses; in TC tasks we denote with $X = \{x_i\}_{i=1}^n$ the set of input texts; in STS we denote with $X = \{x_i = (s_i, s'_i)\}_{i=1}^n$ the set of pairs of input texts to compare. We denote by $f$ the Transformer-based model we aim to explain, and with $f(x_i)$ its prediction on an input $x_i$. Given two tokens $t_a^i, t_b^i$ in an input text $x_i$,[2] we indicate with $\alpha_i(a, b)$ the attention weight of $f$ on the two tokens. We define an *alignment score* by extending the attention weight function to sets of tokens $T_e, T_x$:

$$\alpha_i(T_e, T_x) = \frac{\sum_{a \in T_e, b \in T_x} \alpha_i(a, b)}{\mid T_e \mid + \mid T_x \mid}. \tag{3}$$

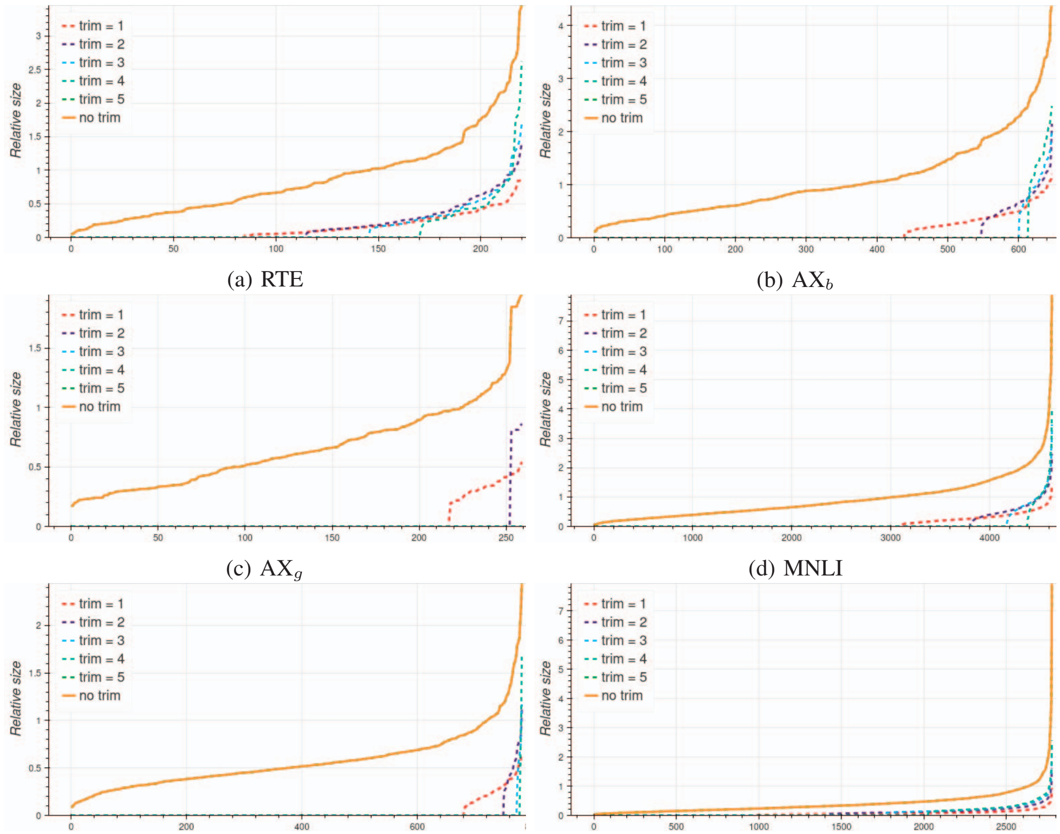
[2]In NLI and STS task, each $x_i$ is given by the concatenation of the two input texts.

Note that, we use the same notation $\alpha(\cdot, \cdot)$ for both tokens or sets of tokens.

Additionally, we address attention weights in a given head $h$ in layer $l$ with $\alpha_i^{h,l}(T_e, T_x)$. We exclusively employ $\alpha$ as an attention, and thus alignment, mechanism between triples and input text spans. In other words, $T_e$ are triples from the explanation, while $T_x$ are tokens from the input text: the hypothesis for NLI tasks, and the whole input text for STS and TC tasks.

*Algorithm:* TRIPLEX begins by extracting a set of triples from the premise $p_i$ (NLI tasks) or the whole input text $t_i$ (STS and TC tasks) through the information extraction module (Line 4), yielding the explanation stub $e$. After removing the duplicate triples from the previous stage, we leverage WordNet to create a set of hypernyms or synonyms (Line 5) for each entity. The function `expand` in Algorithm 1 simply generates the hypernyms/synonyms for the given text according to the task at hand, filtering out the hypothesis in case of an NLI task. We generate the candidate explanations $\widetilde{E}$ by replacing entities in the stub with any of their respective hypernyms (NLI) or synonyms (STS, TC) – Line 5. Then, we compute the hypernym (synonym) distance $d_H$ of each candidate in $\widetilde{E}$ with respect to the stub (Line 11), and return the one at highest hypernym (synonym) distance (Line 15). Finally, we assign to each triple in $e^*$ an optional *alignment score* computed as in Equation 3 (Lines 18 and 21).

We illustrate a TRIPLEX explanation with an example on an NLI task in Table I. Here, we have a long premise on waste management in the city of Cairo, Egypt, and a hypothesis on daily waste production. TRIPLEX extracts five triples with an associated alignment score, which we




(a) RTE (b) $AX_b$

(c) $AX_g$ (d) MNLI

(e) STS (f) AMAZON

Figure 2: Relative TRIPLEX explanation length for complete and trimmed (dashed line) explanation.

can use to rank them according to their alignment with the hypothesis. The highest-scoring one, "4000 tonnes [of waste] are left" on its own appears to contradict the hypothesis. The lowest-scoring one, "Government encourage finance in waste sector" instead does not provide any significant information in either direction.

## IV. EXPERIMENTS

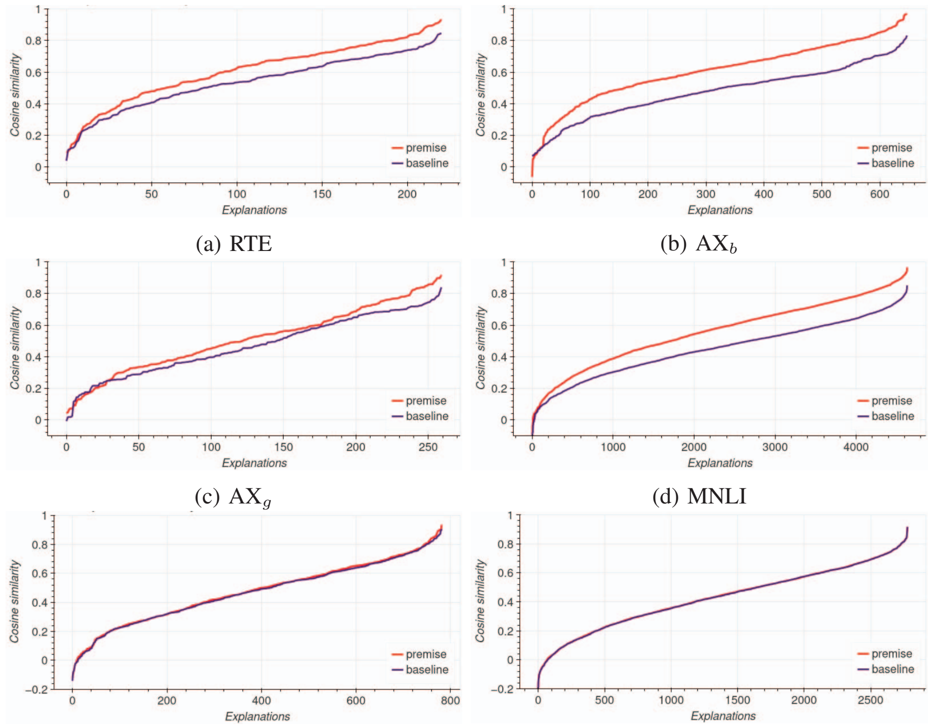
TRIPLEX aims to replace the whole premise/text with a synthesized text that is at the same time concise, understandable, and coherent with the prediction. With this goal in mind, we pose two questions:

Q1: Are TRIPLEX explanations concise?
Q2: Are TRIPLEX explanations coherent?

We aim to answer the former by analyzing explanation complexity and length, and the latter by analyzing explanation similarity with the premise/input text. Label coherence is, as previously stated, a by-product of explanation construction. In this paper we quantitatively answer the two questions with a data-driven analysis, and leave a human evaluation for future work. We evaluate TRIPLEX on four NLI datasets from the GLUE [5] and SuperGLUE [6] benchmarks, namely Recognizing Textual Entailment (RTE) [53], [54], [55], [56], Multi-Genre Natural Language Inference (MNLI) [57], and the $AX_b$ and $AX_g$ diagnostic tasks from [5]; for the STS and TC datasets we evaluate on Amazon Polarity [50] and Semantic Text Similarity [51], respectively. The results are summarized in Table II. Explanations have been extracted from the pre-sampled validation set of each dataset. As a Transformer-based model to explain, we employ the pre-

| | #Records | Performance |
|---|---|---|
| RTE | 276 | 93.0 |
| MNLI | 9814 | 91.4 |
| $AX_g$ | 355 | 92.7 |
| $AX_b$ | 1103 | 53.2 |
| AMAZON | 2530 | 92.8 |
| STS | 1004 | 92.9 |

Table II: Natural Language Inference, Semantic Text Similarity and Text Classification datasets, and DeBERTa performance. Performance is reported on a blind test set, as indicated in the SuperGLUE leader-board and in [50], [51], using accuracy for RTE, MNLI, AMAZON STS, and $AX_g$ and Matthew's correlation for $AX_b$.




(a) RTE (b) $AX_b$

(c) $AX_g$ (d) MNLI

(e) STS (f) AMAZON

Figure 3: Cosine similarity between: *i)* TRIPLEX explanation and INTGRAD baseline; *ii)* TRIPLEX explanation and premise/input text, as computed by SentenceBERT [52].

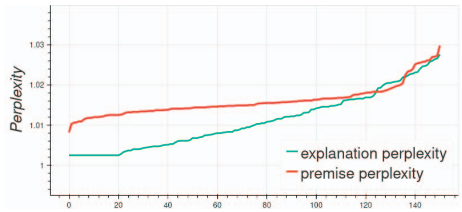
trained DeBERTa [4] model for NLI and STS tasks and the trained Transformer in [50] for the TC task.[3]

We define a set of quantitative evaluation measures and compare TRIPLEX against INTGRAD, a feature importance baseline [58] in which tokens are highlighted in accordance with their importance for the classification. As TRIPLEX explanations correctly predict a label by construction, here we focus on *complexity* and *similarity* measures: the former to evaluate the reduction in complexity obtained by replacing the whole premise/input text with the explanation, and the latter to measure how effective the replacement is. As our goal is to provide simple explanations, we wish to extract small triples while retaining a high similarity with the premise/input text. Moreover, we estimate their likelihood via *perplexity*. Perplexity measures the negative likelihood of a given input text according to a given language model: the better the perplexity, the more natural the text.

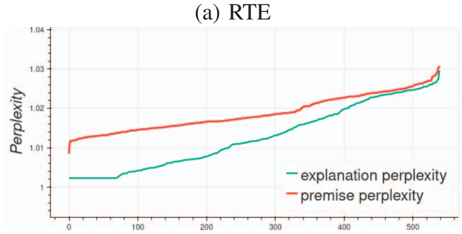
*[Q1] Complexity:* We estimate complexity as "Relative size", i.e., the ratio between the explanation and premise/input text length, both computed in terms of character number. We compute the same indicator on the trimmed explanations, in which we progressively remove triples according to their alignment score. We notice a high complexity in a relatively small number of explanations and a linear increase in the majority, as shown in Figure 2. Values $> 1$ are largely due to short premises/input text and explanations that, once some of its tokens are replaced by longer hypernyms/synonyms, inevitably increase the overall explanation length. Moreover, information extraction techniques tend to decouple and duplicate adjectives so that a phrase `SUBJ ADJ-1 ADJ-2` consisting of only three tokens is expanded into two triples `SUBJ - is - ADJ-1` and `SUBJ - is - ADJ-2`. This behavior is also found in "nested" triples in which a single phrase is expanded into multiple similar triples, significantly increasing the overall explanation complexity. Interestingly, this behavior is less evident on the Amazon dataset (Figure 2 (f)), where longer input text reduces the verbosity and redundancy of the extracted triples. We also report trimmed complexity, in which we trim triples with increasingly higher alignment score. In the plots of Figure 2 we denote with $trim = n$ the number of trimmed triples. Results suggest that trimming triples tends to significantly reduce the complexity of the overall explanations. In other words, the triples with higher alignment score also tend to be the simpler.

*[Q2] Similarity:* Figure 3 reports cosine similarity computed between (a) TRIPLEX explanations and the **baseline** (INTGRAD), and (b) TRIPLEX explanations and the whole

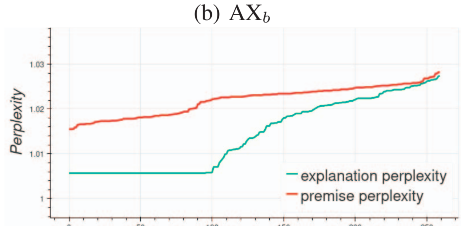
[3]Datasets and model from the huggingface library.



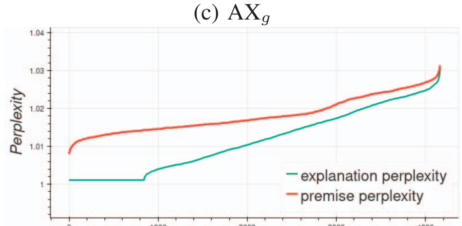

(a) RTE



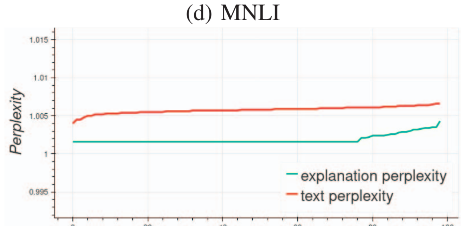

(b) $AX_b$




(c) $AX_g$




(d) MNLI




(e) STS


(f) AMAZON

Figure 4: Explanation and premise/text perplexity.

**premise** or input **text**. Similarities are reported in sorted order. High similarity with the baseline suggests that the extracted triples are indeed relevant to the prediction, while high similarity with the input text suggests that it retains the information in the input text. Most explanations retain a positive similarity, with a linear increase found in all datasets, again due to the direct extraction that we perform. Interestingly, this is also true for the explanation baseline, suggesting that even the keywords identified by the baseline align with the input text and the explanation.

*Perplexity:* We quantitatively estimate the quality of the explanations by means of perplexity [59]. Figure 4 reports sorted perplexity measures on premises/input texts and explanations, as computed by GPT-2 [60]. Perplexities are highly similar, by explanation construction. Since we are directly extracting from the input text with minimum intervention, we expect the extracted propositions to be highly similar in perplexity to the original premise/input text, as we find out to be the case.

## V. Conclusions

In this paper we have introduced TRIPLEX, a post-hoc explanation algorithm for Transformer-based models, with applications to NLI, STS, and TC tasks. Unlike other approaches, we have removed dependence from external text generator models. TRIPLEX explanations show relatively low complexity and are highly similar to baseline existing approaches, yet they meet some limit cases in which explanation complexity degenerates.

As future work, we aim to address such limit cases and to further reduce explanation complexity by properly masking triples that do not contribute to the prediction of the model. We also wish to integrate more complex and commonsense knowledge graphs to better subsume the input text.

## Acknowledgement

This work is partially supported by the European Community H2020 programme under the funding schemes: H2020-INFRAIA-2019-1: Research Infrastructure G.A. 871042 *SoBigData++* (`sobigdata.eu`), G.A. 78835 *Pro-Res* (`prores.eu`), G.A. 761758 *Humane AI* (`humane-ai.eu`), G.A. 825619 *AI4EU* (`ai4eu.eu`), G.A. 952215 *TAILOR* (tailor-network.eu/), and the ERC-2018-ADG G.A. 834756 "XAI: Science and technology for the eXplanation of AI decision making".

## References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[4] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," *arXiv preprint arXiv:2006.03654*, 2020.

[5] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018.

[6] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems," *arXiv preprint arXiv:1905.00537*, 2019.

[7] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 632–642. [Online]. Available: https://www.aclweb.org/anthology/D15-1075

[8] O.-M. Camburu, T. Rocktäschel, T. Lukasiewicz, and P. Blunsom, "e-snli: Natural language inference with natural language explanations," *arXiv preprint arXiv:1812.01193*, 2018.

[9] S. Kumar and P. Talukdar, "Nile: Natural language inference with faithful natural language explanations," *arXiv preprint arXiv:2005.12116*, 2020.

[10] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *arXiv preprint arXiv:1410.5401*, 2014.

[11] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[12] P. M. Htut, J. Phang, S. Bordia, and S. R. Bowman, "Do attention heads in bert track syntactic dependencies?" *arXiv preprint arXiv:1911.12246*, 2019.

[13] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

[14] B. Hoover, H. Strobelt, and S. Gehrmann, "exbert: A visual analysis tool to explore learned representations in transformers models," *arXiv preprint arXiv:1910.05276*, 2019.

[15] S. Jain and B. C. Wallace, "Attention is not explanation," *arXiv preprint arXiv:1902.10186*, 2019.

[16] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *arXiv preprint arXiv:1705.07874*, 2017.

[17] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[18] M. Sundararajan, A. Taly, and Q. Yan, "Gradients of counterfactuals," *arXiv preprint arXiv:1611.02639*, 2016.

[19] T. Lei, R. Barzilay, and T. Jaakkola, "Rationalizing neural predictions," *arXiv preprint arXiv:1606.04155*, 2016.

[20] S. Kim, J. Yi, E. Kim, and S. Yoon, "Interpretation of nlp models through input marginalization," *arXiv preprint arXiv:2010.13984*, 2020.

[21] J. Landthaler, I. Glaser, and F. Matthes, "Towards explainable semantic text matching." in *JURIX*, 2018, pp. 200–204.

[22] H. Chen and Y. Ji, "Learning variational word masks to improve the interpretability of neural text classifiers," *arXiv preprint arXiv:2010.00667*, 2020.

[23] H. Liu, Q. Yin, and W. Y. Wang, "Towards explainable nlp: A generative explanation framework for text classification," *arXiv preprint arXiv:1811.00196*, 2018.

[24] P. A. Jansen, E. Wainwright, S. Marmorstein, and C. T. Morrison, "Worldtree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference," *arXiv preprint arXiv:1802.03052*, 2018.

[25] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 252–262.

[26] C. Bhagavatula, R. L. Bras, C. Malaviya, K. Sakaguchi, A. Holtzman, H. Rashkin, D. Downey, S. W.-t. Yih, and Y. Choi, "Abductive commonsense reasoning," *arXiv preprint arXiv:1908.05739*, 2019.

[27] N. F. Rajani, B. McCann, C. Xiong, and R. Socher, "Explain yourself! leveraging language models for commonsense reasoning," *arXiv preprint arXiv:1906.02361*, 2019.

[28] M. Geva, D. Khashabi, E. Segal, T. Khot, D. Roth, and J. Berant, "Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 346–361, 2021.

[29] B. P. Majumder, O.-M. Camburu, T. Lukasiewicz, and J. McAuley, "Rationale-inspired natural language explanations with commonsense," *arXiv preprint arXiv:2106.13876*, 2021.

[30] J. D. Hwang, C. Bhagavatula, R. L. Bras, J. Da, K. Sakaguchi, A. Bosselut, and Y. Choi, "Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs," *arXiv preprint arXiv:2010.05953*, 2020.

[31] A. Nie, E. D. Bennett, and N. D. Goodman, "Learning to explain: Answering why-questions via rephrasing," *arXiv preprint arXiv:1906.01243*, 2019.

[32] T. Wu, M. T. Ribeiro, J. Heer, and D. S. Weld, "Polyjuice: Automated, general-purpose counterfactual generation," *arXiv preprint arXiv:2101.00288*, 2021.

[33] V. S. Silva, A. Freitas, and S. Handschuh, "Exploring knowledge graphs in an interpretable composite approach for text entailment," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 7023–7030.

[34] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives, "DBpedia: A nucleus for a web of open data," in *ISWC/ASWC*, ser. Lecture Notes in Computer Science, vol. 4825. Springer, 2007, pp. 722–735.

[35] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *WWW*. ACM, 2007, pp. 697–706.

[36] M. Dumontier, A. Callahan, J. Cruz-Toledo, P. Ansell, V. Emonet, F. Belleau, and A. Droit, "Bio2RDF release 3: A larger, more connected network of linked data for the life sciences," in *International Semantic Web Conference (Posters & Demos)*, ser. CEUR Workshop Proceedings, vol. 1272. CEUR-WS.org, 2014, pp. 401–404.

[37] D. S. Himmelstein, A. Lizee, C. Hessler, L. Brueggeman, S. L. Chen, D. Hadley, A. Green, P. Khankhanian, and S. E. Baranzini, "Systematic integration of biomedical knowledge prioritizes drugs for repurposing," *bioRxiv*, 2017. [Online]. Available: https://www.biorxiv.org/content/early/2017/08/31/087619

[38] G. A. Miller, "WORDNET: a lexical database for english," in *HLT*. Morgan Kaufmann, 1992.

[39] N. F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor, "Industry-scale knowledge graphs: lessons and challenges," *Commun. ACM*, vol. 62, no. 8, pp. 36–43, 2019.

[40] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "Swag: A large-scale adversarial dataset for grounded commonsense inference," *arXiv preprint arXiv:1808.05326*, 2018.

[41] N. Mostafazadeh, A. Kalyanpur, L. Moon, D. Buchanan, L. Berkowitz, O. Biran, and J. Chu-Carroll, "Glucose: Generalized and contextualized story explanations," *arXiv preprint arXiv:2009.07758*, 2020.

[42] C. Hao, L. Pang, Y. Lan, Y. Wang, J. Guo, and X. Cheng, "Sketch and customize: A counterfactual story generator," *arXiv preprint arXiv:2104.00929*, 2021.

[43] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu, "Plug and play language models: A simple approach to controlled text generation," *arXiv preprint arXiv:1912.02164*, 2019.

[44] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, "Ctrl: A conditional transformer language model for controllable generation," *arXiv preprint arXiv:1909.05858*, 2019.

[45] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, "Open information extraction from the web," *Communications of the ACM*, vol. 51, no. 12, pp. 68–74, 2008.

[46] Mausam, M. Schmitz, R. Bart, S. Soderland, and O. Etzioni, "Open language learning for information extraction," in *Proceedings of Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL)*, 2012.

[47] K. Gashteovski, R. Gemulla, and L. d. Corro, "Minie: minimizing facts in open information extraction." Association for Computational Linguistics, 2017.

[48] L. Del Corro and R. Gemulla, "Clausie: clause-based open information extraction," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 355–366.

[49] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.

[50] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: understanding rating dimensions with review text," in *Proceedings of the 7th ACM conference on Recommender systems*, 2013, pp. 165–172.

[51] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation," *arXiv preprint arXiv:1708.00055*, 2017.

[52] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[53] I. Dagan, O. Glickman, and B. Magnini, "The PASCAL recognising textual entailment challenge," in *MLCW*, ser. Lecture Notes in Computer Science, vol. 3944. Springer, 2005, pp. 177–190.

[54] R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor, "The second pascal recognising textual entailment challenge," in *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, vol. 6, no. 1. Venice, 2006, pp. 6–4.

[55] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan, "The third pascal recognizing textual entailment challenge," in *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*. Association for Computational Linguistics, 2007, pp. 1–9.

[56] L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo, "The fifth pascal recognizing textual entailment challenge." in *TAC*, 2009.

[57] A. Williams, N. Nangia, and S. R. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *NAACL-HLT*. Association for Computational Linguistics, 2018, pp. 1112–1122.

[58] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3319–3328.

[59] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation," *arXiv preprint arXiv:1603.08023*, 2016.

[60] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.