A Deep Generative XAI Framework for Natural Language Inference Explanations Generation

Anonymous ACL submission

Abstract

Explainable artificial intelligence with natural language explanations (Natural-XAI) aims to produce human-readable explanations as evidence for AI decision-making. This evidence can enhance human trust and understanding of AI systems and contribute to AI explainability and transparency. However, the current approaches focus on single explanation generation only. In this paper, we conduct experiments with the state-of-the-art Transformer architecture and explore *multiple explanations generation* using a public benchmark dataset, e-SNLI (Camburu et al., 2018). We propose a novel deep generative Natural-XAI framework: **INITIATIVE**, standing for *explaIn aNd predIcT wIth contextuAl condiTIonal Variational autoEncoder* for generating natural language explanations and making a prediction at the same time. Our method achieves competitive or better performance against the state-of-the-art baseline models on generation (4.7% improvement in the BLEU score) and prediction (4.4% improvement in accuracy) tasks. Our work can serve as a solid deep generative model baseline for future Natural-XAI research. Our code will be publicly available on GitHub upon paper acceptance.

1 Introduction

With the advancement of modern AI techniques (LeCun et al., 2015), their ubiquitousness comes at the expense of interpretability. Hence, concerns have been raised on whether modern AI can make reasonable judgements (McAllister et al., 2017; Challen et al., 2019), which further triggered an increasing interest in Explainable Artificial Intelligence (XAI) research (Arrieta et al., 2020).

Traditionally, natural language processing (NLP) models are built based on techniques that are inherently more explainable. Examples of such approaches are often referred to as *'white box'* techniques, including rule-based heuristic systems, decision trees, hidden Markov models, conditional random fields, etc. In recent years, due to the advancement of data-driven modelling tools and the big-data era, a *'black box'* technique, deep neural networks have become the dominant approach for modern NLP applications (Danilevsky et al., 2020).

On applying XAI techniques to NLP applications, researchers first focused on *feature*-based- (explanation via important features) (Voskarides et al., 2015; Godin et al., 2018), *model*-based- (explanation via surrogate models) (Ribeiro et al., 2016) and *example*-based (explanation via similar examples) (Croce et al., 2019) explanation techniques. However, even for experts working as data scientists in industry, interpreting results from these models was found to be hard, and bias-prone (Kaur et al., 2020). To reduce human interpretation bias, directly generating natural language explanations seems a better medium for presentation.

This lead to XAI with natural language explanations (or *Natural-XAI*), first proposed in (Camburu et al., 2018), together with a dataset (e-SNLI), extending the Stanford natural language inference (SNLI) dataset (Bowman et al., 2015). Natural language inference (NLI) is the task of determining whether a 'hypothesis' is true (entailment), false (contradiction), or undetermined (neutral) given a 'premise'. NLI is an essential yet challenging task in the natural language understanding field. It requires common sense reasoning on the semantic relationships between premise and hypothesis sentence-pairs. However, as (Gururangan et al., 2018) shows, current NLI datasets contain annotation artefacts, allowing the models to make predictions based on spurious correlations in data. A simple neural network (here a *fastText* classifier (Joulin et al., 2016)) can make correct predictions 67% of the time, when only having access to the hypothesis. However, using the same information, (Camburu et al., 2018) explained that spurious correlations are much harder to be picked up from data when generating explanations, other than predict-

ing the correct label.

Initially, a sequence-to-sequence (*seq2seq*) learning framework was adopted for single-explanation generation (Camburu et al., 2018). When the beam search algorithm is applied, the algorithm can not produce multiple variations of sentences in a principled way (as the top k variations of the beam search list will be qualitatively worse than the first ranked variation) (Gupta et al., 2018). However, the same semantic content can often be expressed in various correct forms in natural language. Hence, this paper adopts deep generative models, to generate multiple high-quality explanations via posterior analysis in the latent space. Additionally, this paper explores how to perform multiple explanations generation, while also making predictions.

Our main contributions include: *(i)* a novel deep generative Natural-XAI framework, **INITIATIVE**, which can generate multiple instances of natural language explanations while making predictions; *(ii)* the first study on spurious correlation on the e-SNLI dataset with Transformer architecture; *(iii)* the first study on the Natural-XAI task with deep generative models; *(iv)* demonstrating the benefits of our framework, **INITIATIVE**, against the state-of-the-art baseline models with empirical experiments; *(v)* a solid deep generative model baseline for future Natural-XAI research.

2 Related Work

2.1 Explainable Artificial Intelligence for Natural Language Processing

General XAI approaches can be categorised in two main ways: (Guidotti et al., 2018; Tjoa and Guan, 2020): 1) Local vs Global, and 2) Self-Explaining vs Post-Hoc. Our work contributes to explainable artificial intelligence (XAI) from two perspectives: *Local* and *Self-Explaining*, as we provide explanations based on (fine-granularity) individual input, and our explanations are directly interpretable.

In terms of explanation techniques and their applications to NLP there are, in general, five different types (Danilevsky et al., 2020): 1) feature importance, 2) surrogate model, 3) example-driven, 4) provenance-based, and 5) declarative induction. The first three are more widely adopted and have already been described briefly in section 1. The provenance-based technique refers to visualising some or all of the prediction process, such as in (Zhou et al., 2018; Amini et al., 2019). Our work uses the *declarative induction technique*, which tackles the challenging task of providing human-readable representations as a part of the results, such as in (Camburu et al., 2018; Pröllochs et al., 2019). Our work further extends (Camburu et al., 2018) with a *probabilistic treatment*. We introduce a novel deep generative framework for *multiple explanation generation and label prediction, simultaneously*.

2.2 Supervised Deep Generative Models in Natural Language

Our work is associated with deep generative models, which is based on neural variational inference (NVI) (Kingma and Welling, 2013; Mnih and Gregor, 2014; Rezende et al., 2014). NVI is also known as amortised variational inference in the literature and can be considered as an extension of the mean-field variational inference (Jordan et al., 1999; Bishop, 2006). NVI technique uses data-driven neural networks instead of more restrictive statistical inference techniques. NVI allows us to infer unobservable latent random variables that generate the observed data and are very efficient for data with hidden structures, such as natural language.

NVI has been successfully applied in various NLP applications including topic modelling (Miao et al., 2016; Srivastava and Sutton, 2017), machine translation (Su et al., 2018; Pagnoni et al., 2018), text classification (Miao et al., 2016), conversation generation (Zhao et al., 2017; Gao et al., 2019), and story generation (Fang et al., 2021). This paper explores the *potential for Natural-XAI explanation generation via building a novel deep generative framework*. This paper is the *first work to apply NVI for the Natural-XAI task*, to the best of our knowledge.

3 Technical Background

This section provides a brief overview of the Conditional Variational Autoencoder (CVAE) and the Transformer architecture. Further, we define our problem to be solved associated with the e-NLI dataset.

3.1 Conditional Variational Autoencoder

CVAE (Sohn et al., 2015; Larsen et al., 2016) is an extended version of the deep generative latent variable model (LVM) based on the variational autoencoder (VAE) model (Kingma and Welling, 2013; Rezende et al., 2014). Both the models allow

learning rich, nonlinear representations for high-dimensional inputs. When compared with VAE (performing inferences for the latent representation $\boldsymbol{z}$, based on the input $\boldsymbol{x}$, only), CVAE performs inference for the latent representation $\boldsymbol{z}$, based on **both** the input $\boldsymbol{x}$ and the output $\boldsymbol{y}$, together. CVAE can be considered as a neural network framework based on supervised neural variational inference.

Compared with a standard autoencoder (Goodfellow et al., 2016), which learns a deterministic mapping from input $\boldsymbol{x}$ to the latent space $\boldsymbol{z}$, CVAE learns the posterior distribution for the latent space $\boldsymbol{z}$, thus allowing sampling from $\boldsymbol{p(z)}$ and interpolation between two points, if they both come from $\boldsymbol{p(z)}$.

CVAE generally includes two components: an encoder and a decoder. We consider the joint probability distribution and its factorisation, in the form of $\boldsymbol{p_\theta(y,z|x) = p_\theta(y|z,x)p_\theta(z|x)}$ as in (Miao et al., 2016; Zhao et al., 2017; Pagnoni et al., 2018; Gao et al., 2019; Fang et al., 2021). The encoder $\boldsymbol{p_\theta(z|x)}$ takes the observed input $\boldsymbol{x}$ and produces a corresponding latent vector $\boldsymbol{z}$ as the output with parameter θ. The decoder $\boldsymbol{p_\theta(y|z,x)}$ takes the observed input $\boldsymbol{x}$ and its corresponding latent vector sample $\boldsymbol{z}$ as the total input and produces an output $\boldsymbol{y}$ with the parameter θ. The latent variable $\boldsymbol{z}$ in the joint probability $\boldsymbol{p_\theta(y,z|x)}$ can be marginalised out by taking samples from $\boldsymbol{p(z)}$.

For CVAE, we optimise the following evidence lower bound (ELBO) for the log-likelihood during training:

$$\begin{aligned} \log p_\theta(y|x) &\geq \mathcal{L}(ELBO) \\ &= E_{q_\phi(z)}[\log p_\theta(y|z,x)] \\ &- D_{KL}[q_\phi(z|x,y)||p_\theta(z|x)] \end{aligned} \tag{1}$$

The first term of ELBO is the reconstruction loss and is measured via cross-entropy matching between predicted versus real target $\boldsymbol{y}$. The second term is the Kullback–Leibler (KL) divergence between two distributions $\boldsymbol{p_\theta(z|x)}$ and $\boldsymbol{q_\phi(z|x,y)}$. As the true posterior distribution $\boldsymbol{p_\theta(z|x)}$ is intractable to compute, a variational family distribution $\boldsymbol{q_\phi(z|x,y)}$ is introduced as its approximation. We consider both $\boldsymbol{p_\theta(z|x)}$ and $\boldsymbol{q_\phi(z|x,y)}$ are in the form of isotropic Gaussian distributions, as $\boldsymbol{\mathcal{N}(\mu_\theta(x), \mathrm{diag}(\sigma^2_\theta(x)))}$ and $\boldsymbol{\mathcal{N}(\mu_\phi(x,y), \mathrm{diag}(\sigma^2_\phi(x,y)))}$. Our work takes a similar assumption, but the key difference lies in the design of our novel model architectures (section 5), together with using the state-of-the-art Transformer model (Vaswani et al., 2017) as a building block. We provide a detailed explanation of the Transformer model in the next section.

3.2 Transformer Architecture

The Transformer architecture was first proposed in (Vaswani et al., 2017) and was the first neural network architecture entirely built based on the self-attention mechanism. It has been used as the main building block for most of the current state-of-the-art models in NLP, such as BERT (Devlin et al., 2018), GPT3 (Brown et al., 2020), and BART (Lewis et al., 2019). The Transformer architecture can be divided into three main components: an embedding part, an encoder and a decoder.

The embedding part takes the input $x \in R^{s_1 \times 1}$ in the form of a sequence with length s_1 and uses an input embedding to create $E(x) \in R^{s_1 \times E}$, where E is the embedded dimension size. Due to the permutation-invariant self-attention mechanism, (Vaswani et al., 2017) further introduced positional encoding, to encode sequential order information, as $P(x) \in R^{s_1 \times E}$. The sum of positional encoding and input embedding is used as the final embedding of the input x. In (Vaswani et al., 2017), sine and cosine functions of different frequencies are adopted as the positional encoding method. Further work for the state-of-the-art large transformers, such as BERT, GPT3 and BART, used a *learnt positional embedding*, which we utilise in this paper.

For the encoder and decoder, we use precisely the same Transformer architecture as in the original paper (Vaswani et al., 2017). We use the official implementation in the Pytorch library[1]. Because the use of Transformers has become common and our implementation is almost identical to the original, we will omit a detailed background description of the model architecture and refer readers to (Vaswani et al., 2017). In our experiments, if an encoder and a decoder are used simultaneously, they each have a separate embedding part.

3.3 Problem Description

Our training data is in the form of N data quadruplets $\{x_n^{(p)}, x_n^{(h)}, y_n^{(l)}, y_n^{(e)}\}_{n=1}^N$, with each quadruplet consisting of the *premise* (denoted by $x_n^{(p)}$), the *hypothesis* (denoted by $x_n^{(h)}$) and their *associated label* (denoted by $y_n^{(l)}$) and *explanation* (de-

[1] https://pytorch.org/docs/stable/nn.html#transformer-layers

noted by $y_n^{(e)}$). For the n^{th} quadruplet, $x_n^{(p)} = \{w_1^{(p)}, ..., w_{L_p}^{(p)}\}$, $x_n^{(h)} = \{w_1^{(h)}, ..., w_{L_h}^{(h)}\}$, $y_n^{(l)} = \{w^{(l)}\}$, and $y_n^{(e)} = \{w_1^{(e)}, ..., w_{L_e}^{(e)}\}$ denote the set of L_p words from the premise sentence, L_h words from the hypothesis sentence, a single word $w^{(l)}$ from the label, and L_e words from the explanation sentence, respectively.

Our validation and testing data are similar to data quadruplets as the training data; however, we have three ($\boldsymbol{y}^{(e_1)}$, $\boldsymbol{y}^{(e_2)}$ and $\boldsymbol{y}^{(e_3)}$) instead of one explanation $\boldsymbol{y}^{(e)}$, created by human experts. During training, we update model parameters based on one explanation $\boldsymbol{y}^{(e)}$; and during validation and testing, we perform model selection and inference based on the mean average loss of three explanations ($\boldsymbol{y}^{(e_1)}$, $\boldsymbol{y}^{(e_2)}$ and $\boldsymbol{y}^{(e_3)}$). In the following descriptions, we will omit the data quadruplet index n and use bold characters to represent vector form representations, as $\boldsymbol{x}^{(p)}$, $\boldsymbol{x}^{(h)}$, $\boldsymbol{y}^{(l)}$, and $\boldsymbol{y}^{(e)}$. These representations will be learnt in an end-to-end fashion.

4 Preliminary Experiments

We present two preliminary experiments in this section. In the first experiment, we select a suitable Transformer architecture from two candidates and explore how easily the Transformer model can capture spurious correlations from data. The second experiment explores how much we can reduce spurious correlations from data, when using explanation as output, other than the label. Additionally, we compare the performance of explanation-generation in full and agnostic scenarios (section 4.2).

For all of our experiments, we use the architecture setting similar to the *base* version of the Transformer model (Vaswani et al., 2017). We use a 6-layer model with 512 hidden units and 8 heads for encoder and decoder networks. Based on an inspection of token length statistics (Appendix A), we set the maximum length of 25 for positional encoding. See Appendix F for a detailed description of all model complexity in this paper.

We generally follow the vocabulary processing steps as in (Camburu et al., 2018) (see detailed pre-processing description in Appendix A) and replace words that appeared less than 15 times with '$< unk >$'. We append '$< bos >$' and '$< eos >$' at the beginning and the end of each sentence during the pre-processing. We report our experiments based on 3 random seeds (1000, 2000 and 3000) and report the average performance with its standard deviation in parenthesis.

We use the maximum a posteriori (MAP) estimation decoding for the conditional generation. MAP decoding, whilst not always the optimal choice, however, has a reasonably good performance, is widely adopted and cheap to compute (Eikema and Aziz, 2020). For the network optimisation, we use Adam (Kingma and Ba, 2014) as our optimiser with default hyper-parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$). We conduct all the experiments with a batch size of 16 and a learning rate of $1e-5$ for a total of 10 epochs on a machine with Ubuntu operating system and a GTX 2080Ti GPU.

4.1 Transformer Architecture Selection and Spurious Correlation Experiments

In the first experiment, we wish to answer two questions: **Q(i)** *What is a good Transformer model architecture choice for the e-SNLI text classification task?* **Q(ii)** *How easily can a Transformer model pick up the spurious correlation, when only a hypothesis sentence is observed?*

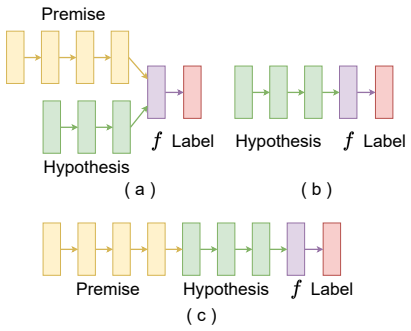

Figure 1: Graphical overview of architectures used in section 4.1. (a) is for Separate Transformer Encoder; (b) is for Premise Agnostic Encoder; and (c) is for Mixture Transformer Encoder.

To answer **Q(i)**, we experiment on two candidate model architectures: *(1) Separate Transformer Encoder*: an architecture with two separate encoders, each for premise and hypothesis sentence (Fig. 1a) *(2) Mixture Transformer Encoder*: an architecture with a mixture encoder for both premise and hypothesis sentence together (Fig 1c). We choose these two candidates for the following reasons: the first candidate architecture is widely adopted in early NLI literature (Parikh et al., 2016; Chen et al., 2017; Gong et al., 2017), where f here refers to algorithmic operations (identity, subtraction, multiplication) as in (Conneau et al., 2017). The latter candidate architecture is adopted by the BERT model (Devlin et al., 2018), where f here refers to

an affine transformation operation and has achieved state-of-the-art performance for NLI tasks. To answer **Q(ii)**, we perform the premise-agnostic prediction experiment on the *Premise Agnostic Encoder* model (Fig 1b), where f here refers to an affine transformation operation.

For the above two experiments, results are presented in Table 1. For the *Separate Transformer Encoder*, we use the encoder outputs at two separate '$< bos >$' positions for algorithmic operations (identity, subtraction and multiplication). For *Mixture Transformer Encoder* and *Premise Agnostic Encoder*, we use the output at the first '$< bos >$' position. We apply an affine transformation operation for predicting the label. The results suggest the *Mixture Transformer Encoder* outperforms the *Separate Transformer Encoder* in a statistically significant way ($p < .05$; Wilcoxon test). The *Premise Agnostic Encoder* achieves 82.84% (based on $65.43/78.98$) of the *Mixture Transformer Encoder* performance, suggesting that Transformer models tend to capture spurious correlations very easily for NLI label prediction task.

Model	Accuracy (%)
Separate Transformer Encoder	73.97 (0.34)
Mixture Transformer Encoder	78.98 (1.44)
Premise Agnostic Encoder	65.43 (0.72)

Table 1: Architecture Selection and Spurious Correlation Experiments.

4.2 Premise-Agnostic and Full Generation Experiments

In the second experiment, we address two further questions: **Q(iii)** *Is providing explanations as output reducing the impact of spurious correlation in a Transformer model, compared to predicting the label only?* **Q(iv)** *How much better are explanations based on premise and hypothesis, instead of hypothesis-only?*

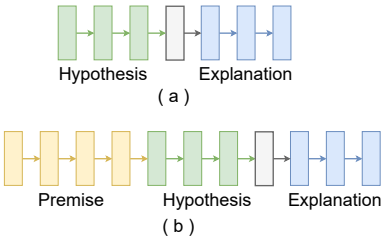

Figure 2: Graphical overview of architectures used in section 4.2. (a) is for Agnostic Generation; (b) is for Full Generation.

To answer **Q(iii)**, we follow and extend the *'PremiseAgnostic'* experiment (Camburu et al., 2018). We use the model architecture in Fig. 2a, and we are interested in evaluating how well the Transformer architecture can generate an explanation from the premise-agnostic scenario (only premise observed). To answer **Q(iv)**, we implement a standard seq2seq model (Sutskever et al., 2014) with Transformer architecture. We compare the agnostic generation scenario with the full generation scenario (both premise and hypothesis observed), the model architecture for complete information is provided in Fig. 2b.

We evaluate the performance of these two models based on both quantitative and qualitative assessments. For qualitative one, we follow (Camburu et al., 2018) and evaluate based on the first 100 test examples only[2] (Correct@100 in Table 2). The qualitative results are calculated based on the highest BLUE score among all three seeds (see details in Appendix B and C). For the quantitative one, we use automatic evaluation metrics (Perplexity and BLEU (Papineni et al., 2002)) over the entire test data points. For evaluation, the lower the perplexity, the higher the BLEU score and the higher the Correct@100, the better the model performs.

Our results, presented in Table 2, suggest that agnostic generation significantly reduces the ability to generate correct explanations, with only 56.9% (based on $35.0/61.5$) for matching words and 26.8% (based on $11/41$) for correctness, based on the first 100 test examples (compared with 82.84% in section 4.1). Selected examples are presented in Appendix D.

Model	Perplexity	BLEU	Correct@100
Agnostic Generation	7.66 (0.03)	25.74 (0.8)	$35.0/11/-$
Full Generation	5.53 (0.05)	33.14 (0.5)	$61.5/41/-$

Table 2: Premise Agnostic Generation Experiments.

5 Deep Generative Natural-XAI Framework for NLI

Our novel deep generative framework consists of two components: an *explanation generative model* and a *label predictive model* (Fig. 3). The generative model uses a novel contextual conditional variational autoencoder (ConCVAE), based on a Transformer-based encoder-decoder architecture.

[2]The three scores are related to matching words, correctness and multiple generations.

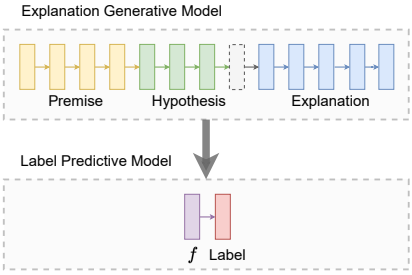

Figure 3: Graphical overview of our Natural-XAI framework, **INITIATIVE**, introduced in section 5.

The predictive model shares the same Transformer encoder parameters with the generative model. Our framework can generate multiple explanations and make prediction, given a pair of premise and hypothesis sentence-pair. In this section, we explain our framework, called **INITIATIVE**, standing for *expla**I**n a**N**d pred**I**c**T** w**I**th contextu**A**l condi**TI**onal **V**ariational auto**E**ncoder*, in detail.

5.1 Neural Encoder

Given a pair of premise $\boldsymbol{x}^{(p)}$ and hypothesis $\boldsymbol{x}^{(h)}$, with their associated explanation $\boldsymbol{y}^{(e)}$, the encoder network outputs two sequences of representations:

$$\begin{aligned} \boldsymbol{x_h} &= \boldsymbol{Encoder}([\boldsymbol{x}^{(p)}; \boldsymbol{x}^{(h)}]) \\ \boldsymbol{y_h} &= \boldsymbol{Encoder}([\boldsymbol{y}^{(e)}]) \end{aligned} \quad (2)$$

Here $\boldsymbol{Encoder}$ refers to the *Transformer Mixture Encoder*, which we selected based on experiments in section 4.1. $\boldsymbol{x_h}$ is the contextual representations for the premise $\boldsymbol{x}^{(p)}$ and hypothesis $\boldsymbol{x}^{(h)}$ pair. $\boldsymbol{y_h}$ is the contextual representation for explanation $\boldsymbol{y}^{(e)}$. We share the same encoder network parameters for producing $\boldsymbol{x_h}$ and $\boldsymbol{y_h}$. $\boldsymbol{x_h}$ has the same sequence length as the sum of premise and hypothesis length. $\boldsymbol{y_h}$ has the same sequence length as the explanation length. $[\boldsymbol{a}; \boldsymbol{b}]$ refers to the concatenation operation of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

5.2 Neural Inferer

The neural inferer can be divided into two separate components: the prior and the posterior networks. Both prior and posterior distributions are assumed to be isotropic multivariate Gaussians, following the CVAE assumption (Sohn et al., 2015; Larsen et al., 2016). As determined by the ELBO equation 1, the parameters of the prior are computed by the prior network, which only takes the inputs: premise $\boldsymbol{x}^{(p)}$ and hypothesis $\boldsymbol{x}^{(h)}$. The posterior parameters are determined from both inputs and outputs: premise $\boldsymbol{x}^{(p)}$, hypothesis $\boldsymbol{x}^{(h)}$ and explanation $\boldsymbol{y}^{(e)}$. We restrict the variance matrices of the prior and the posterior distributions to be diagonal.

5.2.1 Contextual Convolutional Neural Encoder

Before introducing the neural prior and posterior, we first present our novel approach of dealing with various lengths of output from the Transformer encoder. We first adopt the 2d-convolution operations (over the sequence length and hidden dimension) as in (Kim, 2014) and apply it directly to the encoded outputs $\boldsymbol{x_h}$ and $\boldsymbol{y_h}$. For the convolution operations, we use learnable filters with size of 1, 2 and 3 to represent '*unigram*', '*bigram*' and '*trigram*' contextual information from the sequences. Then we use a max-pooling operation over each filter output, to alleviate various sequence-length issue and concatenate them as one single output vector. Finally, we apply an affine transformation on the output vector and return the original vector dimension, but with a sequence length of 1. We name the whole operations here as **c**ontextual c**o**nvolutional **n**eural en**coder** (denoted as $\boldsymbol{Concoder}$).

In contrast, a standard CVAE model takes a fixed position (usually the last hidden state from the sequence, if implemented in the LSTM network) to deal with various sequence-length issues. We implement a standard CVAE with the $< bos >$ position output as the final output, denoted as *CVAE Generation*. We use this as a comparison with our novel solution ($\boldsymbol{Concoder}$), denoted as *ConCVAE Generation* (with results shown in Table 3).

5.2.2 Neural Prior

The prior distribution, denoted as:

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}), \text{diag}(\boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\boldsymbol{x}))) \quad (3)$$

$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ is an isotropic multivariate Gaussian with mean and variance matrices parameterised by neural networks. With variable-length sentence as input, we first use a contextual convolutional neural network, introduced in section 5.2.1, to retrieve a fixed output $\boldsymbol{x_c}$. Then apply two additional affine transformations, $\boldsymbol{f_1}$ and $\boldsymbol{f_2}$, to parameterise the mean and variance matrices for the neural prior. The $\boldsymbol{tanh}()$ function here introduces additional non-linearity and also contributes to numerical stability during parameters optimisation. Thus, we

have:

$$\begin{aligned} \boldsymbol{x_c} &= \boldsymbol{Concoder}([\boldsymbol{x_h}]) \\ \boldsymbol{\mu_\theta} &= \boldsymbol{f_1}([\boldsymbol{x_c}]) \\ \log \boldsymbol{\sigma_\theta} &= \boldsymbol{tanh}(\boldsymbol{f_2}([\boldsymbol{x_c}])) \end{aligned} \tag{4}$$

5.2.3 Neural Posterior

During training, the latent variable will be sampled from the posterior distribution:

$$\boldsymbol{q_\phi}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{\mathcal{N}}(\boldsymbol{z}|\boldsymbol{\mu_\phi}(\boldsymbol{x},\boldsymbol{y}), \text{diag}(\boldsymbol{\sigma_\phi^2}(\boldsymbol{x},\boldsymbol{y}))) \tag{5}$$

$\boldsymbol{q_\phi}(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})$ is also an isotropic multivariate Gaussian with mean and variance matrices parameterised by neural networks. However, the parameters are inferred based on both inputs and outputs. We use the same $\boldsymbol{Concoder}$ network to handle the various length of inputs and outputs ($\boldsymbol{x^{(p)}}$, $\boldsymbol{x^{(h)}}$, and $\boldsymbol{y^{(e)}}$). Similarly, as for the neural prior, we apply two additional affine transformations, $\boldsymbol{f_3}$ and $\boldsymbol{f_4}$, to parameterise the mean and variance matrices. Thus, we have:

$$\begin{aligned} \boldsymbol{y_c} &= \boldsymbol{Concoder}([\boldsymbol{y_h}]) \\ \boldsymbol{\mu_\phi} &= \boldsymbol{f_3}([\boldsymbol{x_c};\boldsymbol{y_c}]) \\ \log \boldsymbol{\sigma_\phi} &= \boldsymbol{tanh}(\boldsymbol{f_4}([\boldsymbol{x_c};\boldsymbol{y_c}])) \end{aligned} \tag{6}$$

5.3 Neural Decoder

The decoder models the probability of the explanation $\boldsymbol{y^{(e)}}$ in an auto-regressive manner, given the predicted label $\boldsymbol{y_p}$, the encoded premise and hypothesis pair $\boldsymbol{x_h}$, and the latent vector $\boldsymbol{z}$. We obtain the explanation sequence via:

$$\boldsymbol{y^{(e)}} = \boldsymbol{Decoder}([\boldsymbol{z};\boldsymbol{x_{(h)}}]) \tag{7}$$

Here, the $\boldsymbol{Decoder}$ refers to the Transformer decoder. Given an explanation with a total sequence length of T, at time step j ($j < T$), it produces the j^{th} word with a softmax selection from the vocabulary based on all the past $j-1$ words.

5.4 Neural Predictor

In our **INITIATIVE** framework, the label can be predicted based on one of the three options: *(i)* **M1 Model**: predicted based on the premise and hypothesis only. *(ii)* **M2 Model**: predicted based on the explanation only. *(iii)* **M3 Model**: predicted based on the premise, hypothesis and explanation all together. With the transformer architecture, we first concatenate the vector outputs of the information at each first '$< bos >$' position to a single vector for each model. Then apply an affine transformation operation $\boldsymbol{f}$ to the concatenated vector. We jointly train the neural predictor together with the generative model *ConCVAE*. We compare the performance of these three models in our experiments (Table 3).

6 Experiments

To evaluate our proposed framework **INITIATIVE**, we conduct experiments to compare with our baseline models. We are interested in the following question: **Q(v)** *How can we generate multiple sentences from our **INITIATIVE** framework and predict class labels at the same time?*

6.1 Baseline Models

We define two types of baseline models: *generative model* and *predictive model*. We consider the following works as baseline models:

- seq2seq (*generative model*, our implementation): a sequence to sequence learning framework developed by (Sutskever et al., 2014). We implement it with the Transformer architecture and denote the experiment results as *Full Generation* in Table 3.
- CVAE (*generative model*, our implementation): a strong probabilistic conditional generation framework introduced by (Sohn et al., 2015; Larsen et al., 2016). We implement it with the Transformer architecture and denote results as *CVAE Generation* in Table 3.
- Transformer (*predictive model*, our implementation): a very strong baseline model for NLI task developed by (Vaswani et al., 2017). We denote the experiment results as *Mixture Transformer Encoder* in Table 3.

6.2 Experiment Setup

To evaluate the explanation generative model of our **INITIATIVE** framework, we implement our novel *ConCVAE* model and compare it with the standard CAVE model. We use the MAP decoding over the latent variable during both training and testing. To answer **Q(v)**, we implement the **INITIATIVE M1**, **M2** and **M3** models (as in section 5.4) and compare

Model	Label Accuracy	Perplexity	BLEU	Correct@100
Premise Agnostic Encoder (lower bound)	65.43 (0.72)	–	–	–
Mixture Transformer Encoder (predictive model baseline)	**78.98** (1.44)	–	–	–
Full Generation (generative model baseline, non-probabilistic)	–	5.53(0.05)	33.14 (0.50)	61.5/41/–
CVAE Generation (generative model baseline, probabilistic)	–	7.58 (0.27)	25.70 (1.04)	47.0/32/12.0
ConCVAE Generation (our model, probabilistic)	–	**5.69 (0.03)**	**32.74 (0.09)**	**65.5/50/14.6**
INITIATIVE M1 (our model)	**83.42 (0.31)**	6.73(0.16)	30.46(0.33)	54.5/44/14.2
INITIATIVE M2 (our model)	73.73(1.54)	**5.75 (0.01)**	32.68(0.64)	59.0/42/12.0
INITIATIVE M3 (our model)	79.85(0.35)	5.93(0.02)	**32.70 (0.28)**	**60.5/48/13.8**

Table 3: Natural-XAI explanation Generation Results ('–' refers to results not applicable). We use the same evaluation method for Correct@100 as detailed in Appendix C.

their performance to our predictive and generative baseline models. Regarding neural network architecture, vocabulary and training, we use the same experimental setting as in section 4.

6.3 Interpolation in Latent Space

To generate multiple explanations, we perform posterior analysis over the latent space. We choose to linearly interpolate the isotropic multivariate Gaussians over its 95.44% region (left and right of 2σ from μ). We produce 5 samples calculated based on $\mu - 2\sigma$, $\mu - \sigma$, μ, $\mu + \sigma$, and $\mu + 2\sigma$ coordinates over the latent space. We check if different explanations can be generated with similar semantic meaning, based on the criterion detailed in Appendix C. Qualitative evaluation results for interpolations are presented in the Correct @100 column in Table 3. Examples of interpolation results from the *ConCVAE Generation* experiment are presented in Appendix E.

7 Results and Discussion

The main results are presented in Table 3. For explanation generation evaluation, we compare a deep generative model (*CVAE Generation*) with a standard neural network model (*Full Generation*), with similar model complexity (as in Appendix F). The results suggest that for the *CVAE Generation* model, for quantitative results, the perplexity is increased (2.05), the BLEU score is reduced (7.4%). We obtain a worse score for qualitative assessment in matching words (14.5 less) and correctness (9 less), meaning the performance is worse than the *Full Generation* model. However, deep generative models such as *CVAE Generation* allow generating multiple explanations via a posterior analysis over the latent space. With our novel contextual deep generative model *ConCVAE*, we can achieve competitive performance with the *Full Generation* model, with significant improvements in qualitative results (Correct @100), as shown in Table 3.

We implement three variants of our **INITIATIVE** framework (**M1**, **M2** and **M3**) to perform generation and prediction simultaneously. Results suggest that generating a valid explanation from the premise and hypothesis sentence-pair allows the encoder to better understand the semantics meaning of the words and hence further enhances the accuracy of prediction. This leads to a boost in prediction performance (83.42% for **M1** and 79.85% for **M3**), compared to the Mixture Transformer Encoder (78.98%), with a prediction network with the same number of parameters. However, as shown in the **M2** model, the prediction accuracy is worse when using explanation-only to predict the label. This makes sense, as the best performance generative model (*ConCVAE*) only finds 50% of the correct explanation (based on the correctness score of 50) in the first 100 test examples. We also observe that for the **M3** model, the generation results are much better than for the **M1** model (perplexity decrease of 0.8 and BLEU increase of 2.2%). For Natural-XAI with label prediction and explanation generation together, for prediction performance, the **M1** model fits better. However, for generation performance, the **M3** model fits better with our purpose. Additionally, we observe that label prediction results in decreasing generative performance, as opposed to the *ConCVAE* model.

8 Conclusion and Future Work

In this paper, we present our novel deep generative Natural-XAI framework, **INITIATIVE**. Our framework can generate multiple explanations and predict the label simultaneously, achieving competitive or better performance against the state-of-the-art baseline models on both the generation (4.7% improvement in BLEU) and prediction (4.4% improvement in accuracy) tasks. Our method can serve as a solid baseline for future Natural-XAI research and suggests a more generative perspective for future research in this field.

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115.

Christopher M Bishop. 2006. Pattern recognition. *Machine learning*, 128(9).

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. *arXiv preprint arXiv:1812.01193*.

Robert Challen, Joshua Denny, Martin Pitt, Luke Gompels, Tom Edwards, and Krasimira Tsaneva-Atanasova. 2019. Artificial intelligence, bias and clinical safety. *BMJ Quality & Safety*, 28(3):231–237.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2017. Neural natural language inference models enhanced with external knowledge. *arXiv preprint arXiv:1711.04289*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Danilo Croce, Daniele Rossini, and Roberto Basili. 2019. Auditing deep learning processes through kernel-based explanatory 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 4037–4046.

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A survey of the state of explainable ai for natural language processing. *arXiv preprint arXiv:2010.00711*.

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

Bryan Eikema and Wilker Aziz. 2020. Is map decoding all you need? the inadequacy of the mode in neural machine translation. *arXiv preprint arXiv:2005.10283*.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, Guodong Zhou, and Shuming Shi. 2019. A discrete cvae for response generation on short-text conversation. *arXiv preprint arXiv:1911.09845*.

Fréderic Godin, Kris Demuynck, Joni Dambre, Wesley De Neve, and Thomas Demeester. 2018. Explaining character-aware neural networks for word-level prediction: Do they discover linguistic rules? *arXiv preprint arXiv:1808.09551*.

Yichen Gong, Heng Luo, and Jian Zhang. 2017. Natural language inference over interaction space. *arXiv preprint arXiv:1709.04348*.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. 2020. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *International conference on machine learning*, pages 1558–1566. PMLR.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Rowan McAllister, Yarin Gal, Alex Kendall, Mark Van Der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. 2017. Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *International conference on machine learning*, pages 1727–1736.

Andriy Mnih and Karol Gregor. 2014. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pages 1791–1799. PMLR.

Artidoro Pagnoni, Kevin Liu, and Shangyan Li. 2018. Conditional variational autoencoder for neural machine translation. *arXiv preprint arXiv:1812.04405*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Nicolas Pröllochs, Stefan Feuerriegel, and Dirk Neumann. 2019. Learning interpretable negation rules via weak supervision at document level: A reinforcement learning approach. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 407–413. Association for Computational Linguistics.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " 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*, pages 1135–1144.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. *arXiv preprint arXiv:1703.01488*.

Jinsong Su, Shan Wu, Deyi Xiong, Yaojie Lu, Xianpei Han, and Biao Zhang. 2018. Variational recurrent neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Erico Tjoa and Cuntai Guan. 2020. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Nikos Voskarides, Edgar Meij, Manos Tsagkias, Maarten De Rijke, and Wouter Weerkamp. 2015. Learning to explain entity relationships in knowledge graphs. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 564–574.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *arXiv preprint arXiv:1703.10960*.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. *arXiv preprint arXiv:1801.04726*.

A Dataset Statistics

Model	Mean	Median	Standard Deviation	Min	Max
Premise	17	15	7	4	84
Hypothesis	11	10	4	3	64
Explanation	16	15	7	2	189

Table 4: Token length statistics for the e-SNLI dataset, all numbers round to integer.

Our detailed dataset statistics are presented in Table 4, to help reproduce the experiment results, we provide a detailed description of our pre-processing and tokenisation process. We start by stripping out any space in front of and behind the original sentence. And then tokenise it using the Spacy English tokeniser tool based on the '$en\_core\_web\_sm$' lexicon resource. The tokenised text is then used to create the complete vocabulary for training. We follow (Camburu et al., 2018) and remove tokens that appear less than 15 times. We additional include special tokens '$< unk >$', '$< pad >$', '$< bos >$' and '$< eos >$' in the vocabulary. Before we use each sentence, we append '$< bos >$' at the beginning of this sentence and append '$< eos >$' at the end of this sentence, with a space in between.

B Explanation Template Examples

We provide the following list of explanation template examples as the guidelines to filter out non-informative explanations. Our templates are built based on the templates in (Camburu et al., 2018) and our own generated explanations.

B.1 General Templates

- <premise>
- <hypothesis>

B.2 Contradiction Templates

- <XXX> is either <XXX> or <XXX>
- <XXX> is not the same as <XXX>
- <XXX> can not be both <XXX> and <XXX> at the same time
- <XXX> is not <XXX>
- <XXX> can not <XXX>
- <XXX> is <XXX>, not <XXX>

B.3 Entailment Templates

- <XXX> is the same as <XXX>
- <XXX> is a type of <XXX>
- <XXX> is a <XXX>
- <XXX> is a rephrasing of <XXX>
- <XXX> so <XXX>

B.4 Neutral Templates

- <XXX> does not mean <XXX>
- just because <XXX> does not mean <XXX>
- <XXX> is not necessarily <XXX>
- <XXX> does not imply <XXX>
- not all <XXX> are <XXX>

C Qualitative Evaluation

We provide a detailed qualitative evaluation criterion here used for the first 100 testing examples in this paper. Our evaluation results are calculated based on the best BLEU score among the three runs of the experiments, based on different random seeds (1000, 2000 and 3000). The final results are averaged based on three individuals' opinions. We first filter our the non-informative explanations based on the templates provided in Appendix B and then we evaluate the following aspects:

1. Matching words: we check if the generated explanation contains the key matching words (or phrases) from its associated premise and hypothesis sentence pair (based on three golden references). Each premise and hypothesis sentence is assigned with a 0.5 score (hence a pair of them have a score of 1, and the first 100 examples have a total of 100 score). We give a score of either 0.5 or 0 for each premise or hypothesis sentence. Matching words means no word replacements hence only the exact words taken from the premise and hypothesis sentence are correct. In this case, *'car'* and *'vehicle'* are not matching words. Additionally, partially correct words (or phrases) are considered as incorrect. In this case, *' red car'* and *'yellow car'* are not matching words. However, we accept change in grammatical voice, such as *'walking'* is the same as *'walk'* and grammatical articles such as *'a car'* is the same as *'the car'*.

2. Correctness, we check if the generated explanation can be used as a reasonable and correct explanation for premise and hypothesis sentence pair when we get at least 0.5 score in the matching words check. Each explanation sentence is assigned with a score of 1 (hence we have a total of 100 score for the first 100 examples). We either give a score of 1 or 0 for each explanation sentence based on evaluation against three golden explanation examples.

3. Multiple Generation, we check if the model can generate multiple explanations based on interpolation when we get a 1 score in the correctness check. We produce 5 interpolation results based on the methods in section 6.3, and each resulting sentence is assigned with a 0.2 score (hence we have a total score of 1 for each explanation and a total of 100 score for the first 100 examples). We give either 0.2 or 0 for each explanation sentence based on evaluation against three golden explanation examples. If two of the sentence instances are exactly the same, we consider only one valid. However, we allow missing matching words and correctness for multiple generations as long as semantic the instances are similar.

We present the evaluation results in order, separate with '/'. For the non-probabilistic model, the multiple generations are not applicable and are marked as '–.'

D Generated Explanations

In this section, we present examples taken from the experiments in section 4.2, these examples are from two scenarios *(i)* agnostic experiment where the agnostic generation model can pick up spurious correlation to generate the correct explanations. However, the premise information is not offered. Hence, the explanations generation should ideally be incorrect, as shown in Table 5 *(ii)* agnostic experiment where the agnostic generation model is not able to pick up the spurious correlation. While the full generation model can generate the correct explanations, as shown in Table 6. In the first 100 test examples, case *(i)* happens 11 times and *(ii)* happens 41 times.

E Interpolation Explanations

In this section we presents examples taken from the experiments in section 6 with our model **Con-**

Test Data Number	22
Premise	***one tan girl*** with a wool hat is running and leaning over an object , while another person in a wool hat is sitting on the ground .
Hypothesis	***a boy*** runs into a wall
Explanation 1	there are either two people - a girl and another person - or there is a boy .
Explanation 2	a boy is not a tan girl .
Explanation 3	the person is either a girl or boy , not both
Agnostic Generation	a boy is not a girl .
Full Generation	a boy is not a girl .
Test Data Number	30
Premise	***a couple walk*** hand in hand down a street .
Hypothesis	***a couple is sitting*** on a bench .
Explanation 1	the couple can not be walking and sitting a the same time .
Explanation 2	a couple ca n't both be sitting and walking .
Explanation 3	sitting is not the same as walking .
Agnostic Generation	a couple can not be sitting on a bench and walking down a street at the same time .
Full Generation	the couple can not be walking and sitting at the same time .
Test Data Number	91
Premise	***a dog*** jumping for a frisbee in the snow .
Hypothesis	***a cat*** washes his face and whiskers with his front paw .
Explanation 1	dogs and cats are not the same animal , and they are performing different activities : the dog jumps while the cat engages in cleaning himself .
Explanation 2	a dog is a different from a cat
Explanation 3	a dog is not a cat .
Agnostic Generation	a dog is not a cat .
Full Generation	a dog is not a cat .

Table 5: Selected spurious correlation examples.

TrCVAE, these examples are generated based on the linear interpolation methods presented in section 6.3. We only present the multiple generation results, which are different sentences and omit the same ones.

F Model Complexity

We present the model complexity in Table 8, with separate counts for prediction, generation and total network components, the one with the '–' mark is denoted as not applicable. Since we share the same parameters for the Transformer encoder network in our **EAP-ConTrCVAE** framework, our framework can perform explanation generation and label prediction and keep the same model complexity as the generation model *ConTrCVAE*.

Test Data Number	4
Premise	***a woman*** with a green headscarf , blue shirt and a very big grin .
Hypothesis	the ***woman is young*** .
Explanation 1	the woman could 've been old rather than young
Explanation 2	there is no indication that the woman is young .
Explanation 3	not all women are young .
Agnostic Generation	the woman is young is the same as the woman is young .
Full Generation	not all women are young .
Test Data Number	9
Premise	an ***old man*** with a package ***poses in front of an advertisement*** .
Hypothesis	a ***man walks by an ad*** .
Explanation 1	the man poses in front of the advertisement therefore he did not walk by it .
Explanation 2	a man can not be walking by an ad while posing in front of it .
Explanation 3	poses is different from walks .
Agnostic Generation	a man walks by an ad is the same as a man walks by an ad .
Full Generation	the man either poses or walks by .
Test Data Number	26
Premise	a young ***family*** enjoys ***feeling ocean waves*** lap at their feet .
Hypothesis	a ***family*** is out ***at a restaurant*** .
Explanation 1	one says the family is inside the restaurant , while the other says they are feeling ocean waves .
Explanation 2	if a family enjoys ocean waves they are not out at a restaurant .
Explanation 3	family ca n't be at restraint if feeling ocean waves
Agnostic Generation	a family is at a restaurant is a rephrasing of a family is out at a restaurant .
Full Generation	a family can not be at a restaurant and at the ocean at the same time .
Test Data Number	69
Premise	an older ***women tending to a garden*** .
Hypothesis	the ***lady*** is ***cooking dinner***
Explanation 1	the lady can not be cooking dinner if she is tending to a garden
Explanation 2	a woman can not be gardening and cooking at the same time .
Explanation 3	the woman is either tending to a garden or cooking .
Agnostic Generation	the lady can not be cooking dinner and sitting on a bench at the same time .
Full Generation	the lady can not be tending to a garden and cooking dinner at the same time .
Test Data Number	77
Premise	***a man*** in a black shirt is ***looking at a bike*** in a workshop .
Hypothesis	***a man*** is ***deciding which bike to buy***
Explanation 1	or the man works in the work shop and is selling the bike , or is inspecting the bike , looking for issues rather than deciding .
Explanation 2	just because the man is looking at a bike does n't mean he is deciding which bike to buy .
Explanation 3	the man looking at the bike may not be deciding to buy a bike at all .
Agnostic Generation	a man is not a woman .
Full Generation	looking at a bike does not imply deciding to buy .
Test Data Number	97
Premise	***a girl*** playing ***a violin*** along with a group of people
Hypothesis	***a girl*** is playing ***an instrument*** .
Explanation 1	playing an instrument is another less detailed way to say playing a violin
Explanation 2	the violin is an instrument .
Explanation 3	the violin is an instrument .
Agnostic Generation	a girl is playing an instrument is a rephrasing of a girl is playing an instrument .
Full Generation	a violin is an instrument .

Table 6: Selected none-spurious correlation examples.

Test Data Number	29
Premise	a **couple** walk hand in hand down a street .
Hypothesis	the **couple** is **married** .
Explanation 1	just because the couple is hand in hand does n't mean they are married .
Explanation 2	just because the couple is walking hand in hand does n't mean they 're married .
Explanation 3	the couple walking down the street holding hands does not mean that they are married .
Generated Explanation 1	not all couple walking down street are married .
Generated Explanation 2	not all couple in hand is married .
Generated Explanation 3	not all couples are married .
Test Data Number	50
Premise	a little boy in a gray and white striped sweater and tan pants is **playing** on a piece of **playground equipment** .
Hypothesis	the boy is **sitting on** the school bus on his **way home** .
Explanation 1	school buses normally are not located on playgrounds , so a child can not be playing with playground equipment in a bus .
Explanation 2	the boy is either playing on a piece of playground equipment or sitting on the school bus on his way home .
Explanation 3	there ca n't be a playground on a school bus .
Generated Explanation 1	the boy can not be playing on a playground and sitting on his way home at the same time .
Generated Explanation 2	the boy can not be playing on a playground and sitting on his way home simultaneously .
Generated Explanation 3	the boy can not be playing on a playground and sitting on the bus at the same time .
Test Data Number	64
Premise	people jump over **a mountain** crevasse on a rope .
Hypothesis	people are jumping **outside** .
Explanation 1	the jumping over the mountain crevasse must be outside .
Explanation 2	a mountain crevasses is usually located outside .
Explanation 3	the mountain is outside .
Generated Explanation 1	people jump over a mountain so they must be outside .
Generated Explanation 2	a mountain is outside .
Test Data Number	77
Premise	a man in a black shirt is **looking at** a bike in a workshop .
Hypothesis	a man is deciding which bike **to buy**
Explanation 1	or the man works in the work shop and is selling the bike , or is inspecting the bike , looking for issues rather than deciding .
Explanation 2	just because the man is looking at a bike does n't mean he is deciding which bike to buy .
Explanation 3	the man looking at the bike may not be deciding to buy a bike at all .
Generated Explanation 1	just because a man is looking at a bike in a workshop does n't mean he is deciding to buy .
Generated Explanation 2	just because a man is looking at a bike in a workshop does n't mean he is deciding what to buy .

Table 7: Selected spurious correlation examples.

Model	Prediction	Generation	Total
Separate Transformer Encoder	48.6M	-	48.6M
Mixture Transformer Encoder	24.3M	-	24.3M
Premise Agnostic Encoder	24.3M	-	24.3M
Agnostic Generation	-	63.6M	63.6M
Full Generation	-	63.6M	63.6M
CVAE Generation	-	65.9M	65.9M
ConTrCVAE Generation	-	68.3M	68.3M
EAP-ConTrCVAE M1	24.3M	68.3M	68.3M
EAP-ConTrCVAE M2	24.3M	68.3M	68.3M
EAP-ConTrCVAE M3	24.3M	68.3M	68.3M

Table 8: Number of parameters for each model, with separate counts for prediction and generation component.