Exploiting Inductive Bias in Transformers for Unsupervised Disentanglement of Syntax and Semantics with VAEs

Ghazi Felhi, Joseph Le Roux
LIPN
Université Sorbonne Paris Nord - CNRS UMR 7030
F-93430, Villetaneuse, France
{felhi, leroux}@lipn.fr

Djamé Seddah
INRIA Paris
Paris, France
djame.seddah@inria.fr

Abstract

We propose a generative model for text generation, which exhibits disentangled latent representations of syntax and semantics. Contrary to previous work, this model does not need syntactic information such as constituency parses, or semantic information such as paraphrase pairs. Our model relies solely on the inductive bias found in attention-based architectures such as Transformers.

In the attention of Transformers, *keys* handle information selection while *values* specify what information is conveyed. Our model, dubbed QKVAE, uses Attention in its decoder to read latent variables where one latent variable infers keys while another infers values.

We run experiments on latent representations and experiments on syntax/semantics transfer which show that QKVAE displays clear signs of disentangled syntax and semantics. We also show that our model displays competitive syntax transfer capabilities when compared to supervised models and that comparable supervised models need a fairly large amount of data (more than 50K samples) to outperform it on both syntactic and semantic transfer. The code for our experiments is publicly available[1].

1 Introduction

Disentanglement, a process aimed at obtaining neural representations with identified meaning, is a crucial component of research on interpretability (Rudin et al., 2022). A form of disentanglement that received a lot of interest from the NLP community is the separation between syntax and semantics in neural representations (Chen et al., 2019; Bao et al., 2019; Zhang et al., 2019; Chen et al., 2020; Huang and Chang, 2021; Huang et al., 2021). Previous works perform disentanglement using paraphrase pairs as information for semantics, and/or constituency parses as information for syntax. The dependence of models on labeled data is known to entail high cost (see Seddah et al., 2020 on syntactic annotation), and to often require new labels to handle problems such as concept drift (Lu et al., 2019) and domain adaptation (Farahani et al., 2021).

In light of the above, we propose an unsupervised model which directs syntax and semantics into different neural representations without semantic or syntactic information. In the Transformer architecture (Vaswani et al., 2017), the attention mechanism is built upon a *query* from a set Q, which pools *values* V through *keys* K. For each query, values are selected according to their matching score computed by the similarity between their corresponding keys and the query. Building on an analogy between the (K, V) couple and syntactic roles with their lexical realizations (explicited in §4.2) we present QKVAE[2], a Transformer-based Variational Autoencoder (VAE; Kingma and Welling, 2014).

To build our model, we modify a previous Transformer-based VAE, called the Attention-Driven VAE (ADVAE; Felhi et al., 2021). Using Cross-Attention, our model encodes sentences into two latent variables: z^{sem} to infer values for V, and z^{syn} to assign keys in K for values in V. These keys and values are then used in the Attention mechanism of a Transformer Decoder to generate sentences. We show that z^{syn} tends to contain syntactic information, while z^{sem} tends to represent semantic information. Additionally, comparisons with a supervised model show that it needs a considerable amount of data to outperform our model on syntactic and semantic transfer metrics.

Our contributions can be summarized as follows:

- We describe QKVAE, a model designed to disentangle syntactic information from semantic information by using separate latent variables for keys and values in Transformers Attention.

[1] github.com/ghazi-f/QKVAE

[2] A contraction of the (Q, K, V) triplet with the VAE acronym.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5763 - 5776
July 10-15, 2022 ©2022 Association for Computational Linguistics

- We run experiments on a dataset for English which empirically show that the two types of latent variables have strong preferences respectively for syntax and semantic.
- We also show that our model is capable of transferring syntactic and semantic information between sentences by using their respective latent variables. Moreover, we show that our model's syntax transfer capabilities are competitive with supervised models when they use their full training set (more than 400k sentences), and that a supervised model needs a fairly large amount of labeled data (more than 50k samples) to outperform it on both semantic and syntactic transfer.

2 Related Work

We broadly divide works on explainability in NLP into two research directions. The first seeks *post hoc* explanations for black-box models, and led to a rich literature of observations on the behavior of Neural Models in NLP (Tenney et al., 2019; Jawahar et al., 2019; Hu et al., 2020; Kodner and Gupta, 2020; Marvin and Linzen, 2020; Kulmizev et al., 2020; Rogers et al., 2020). Along with these observations, this line of works also led to numerous advances in methodology concerning, for instance, the use of attention as an explanation (Jain and Wallace, 2019; Wiegreffe and Pinter, 2020), the validity of probing (Pimentel et al., 2020), or contrastive evaluation with minimal pairs (Vamvas and Sennrich, 2021). The second research direction on explainability in NLP seeks to build models that are explainable by design. This led to models with explicit linguistically informed mechanisms such as the induction of grammars (RNNG; Dyer et al., 2016, URNNG; Kim et al., 2019) or constituency trees (ON-LSTM; Shen et al., 2019, ONLSTM-SYD; Du et al., 2020).

Disentangled representation learning is a subfield of this second research direction which aims at separating neural representations into neurons with known associated meanings. This separation was performed on various characteristics in text such as style (John et al., 2020; Cheng et al., 2020), sentiment and topic (Xu et al., 2020), or word morphology (Behjati and Henderson, 2021). In works on disentanglement, consequent efforts have been put in the separation between syntax and semantics, whether merely to obtain an interpretable specialization in the embedding space (Chen et al., 2019; Bao et al., 2019; Ravfogel et al., 2020; Huang et al., 2021), or for controllable generation (Zhang et al., 2019; Chen et al., 2020; Huang and Chang, 2021; Li et al., 2021). However, all these works rely on syntactic information (constituency parses and PoS tags) or semantic information (paraphrase pairs). To the best of our knowledge, our work is the first to present a method that directs syntactic and semantic information into assigned embeddings in the challenging unsupervised setup.

From a broader machine learning perspective, using knowledge of the underlying phenomena in our data, we design our model QKVAE with an inductive bias that induces understandable behavior in an unsupervised fashion. Among the existing line of applications of this principle (Rezende et al., 2016; Hudson and Manning, 2018; Locatello et al., 2020; Tjandra et al., 2021), ADVAE (Felhi et al., 2021), the model on which QKVAE is based, is designed to separate information from the realizations of different syntactic roles without supervision on a dataset of regularly structured sentences.

3 Background

In this section, we go over the components of our model, namely VAEs, attention in Transformers, and ADVAE, the model on which QKVAE is based.

3.1 VAEs as Language Models

Given a set of observations w, VAEs are a class of deep learning models that train a generative model $p_\theta(w) = \int_z p(z)p_\theta(w|z)dz$, where $p(z)$ is a prior distribution on latent variables z that serve as a seed for generation, and $p_\theta(w|z)$ is called the decoder and generates an observation w from each latent variable value z. Since directly maximizing the likelihood $p_\theta(w)$ to train a generative model is intractable, an approximate inference distribution $q_\phi(z|w)$, called the encoder, is used to formulate a lower-bound to the exact log-likelihood of the model, called the Evidence Lower-Bound (ELBo):

$$\log p_\theta(w) \geq \mathbb{E}_{(z)\sim q_\phi(z|w)}\left[\log p_\theta(w|z)\right] - \mathrm{KL}[q_\phi(z|w)||p(z)] = \mathrm{ELBo}(w; z) \quad (1)$$

Early works on VAEs as language models have shown that, contrary to non-generative sequence-to-sequence (Sutskever et al., 2014) models, they learn a smooth latent space (Bowman et al., 2016). In fact, this smoothness enables decoding an interpolation of latent codes (*i.e.* a homotopy) coming from two sentences to yield a well-formed third sentence that clearly shares characteristics (syntactic, semantic...) with both source sentences. This interpolation will be used as a control baseline in our experiments.

3.2 Attention in Transformers.

The inductive bias responsible for the disentanglement capabilities of our model is based on the design of Attention in Transformers (Vaswani et al., 2017). In attention mechanisms, each element of a series of query vectors $Q = \{q_1, \ldots, q_{|Q|}\}$ performs a soft selection of values $V = \{v_1, \ldots, v_{|V|}\}$ whose compatibility with the query is given by their corresponding key vector in $K = \{k_1, \ldots, k_{|V|}\}$ via dot product. For each $q_i \in Q$, the series of dot products is normalized and used as weights for a convex interpolation of the values. Formally, the result is compactly written as:

$$\text{Attention}(Q, K, V) = \text{Softmax}(QK^T)V \quad (2)$$

Here, we stress that K is only capable of controlling what information is selected from V, while V is responsible for the value of this information. Using the above operators and the embedding level concatenation operator Cat, Multi-Head Attention (MHA) in Transformers is defined as follows:

$$\begin{aligned} &\text{MHA}(\tilde{Q}, \tilde{K}, \tilde{V}) = \text{Cat}(head_1, ...head_H)W^O \\ &s.t: \ head_i = \text{Attention}(\tilde{Q}W_i^Q, \tilde{K}W_i^K, \tilde{V}W_i^V) \end{aligned}$$

Where W^O, W_i^Q, W_i^K, and W_i^V are trainable parameter matrices. In turn, Self-Attention (SA) and Cross-Attention (CA) are defined, for sets of elements called source S and target T, as follows:

$$\begin{aligned} \text{SA}(T) &= \text{MHA}(T, T, T) \\ \text{CA}(T, S) &= \text{MHA}(T, S, S) \end{aligned}$$

The above SA mechanism is used to exchange information between elements of target T, while in CA, targets T pull (or *query* for) information from each element of the source S. Transformer Encoders (Enc) are defined as the composition of layers each consisting of an attention followed by a Feed-Forward Network F:[3]

$$\text{Enc}(T) = \tilde{T}_{D^{enc}}, \text{ s.t. } \tilde{T}_d = \begin{cases} T \text{ if } d = 0, \text{ else:} \\ \text{F}(\text{SA}(\tilde{T}_{d-1})) \end{cases}$$

[3]We omit residual connections and layer normalizations after each SA or CA for simplicity.

Transformer Decoders (Dec) are defined with instances of SA, CA and F:

$$\text{Dec}(T, S) = \tilde{T}_{D^{dec}}, \text{s.t. } :$$

$$\tilde{T}_d = \begin{cases} T \text{ if } d = 0, \text{ else:} \\ \text{F}(\text{CA}(\text{SA}(\tilde{T}_{d-1}), S)) \end{cases}$$

where D^{enc} and D^{dec} above are respectively the number of layers of Enc and Dec. For autoregressive decoding, Vaswani et al. (2017) define a version of Dec we will call $\overline{\text{Dec}}$. In this version, the result of each QK^T (Eq. 2) in Self-Attention is masked so that each t_i in T only queries for information from t_j with $j \leq i$. Even though $\overline{\text{Dec}}$ yields a sequence of length equal to that of target T, in the following sections we will consider its output to be only the last element of $\tilde{T}_{D^{dec}}$ in order to express auto-regressive generation in a clear manner.

3.3 ADVAE

ADVAE is a Variational Autoencoder for unsupervised disentanglement of sentence representations. It mainly differs from previous LSTM-based (Bowman et al., 2016) and Transformer-based (Li et al., 2020b) VAEs in that it uses Cross-Attention to encode and decode latent variables, which is the cornerstone of our model. In ADVAE, Cross-Attention is used to: *i)* encode information from sentences into a fixed number of vectorial latent variables; *ii)* decode these vectorial latent variables by using them as sources for the target sentences generated by a Transformer Decoder.

Formally, let us define M^μ, M^σ, and M^w to be linear layers that will respectively be used to obtain the latent variables' means and standard deviations, and the generated words' probabilities, L the number of vectorial latent variables $z = \{z_1, \ldots, z_L\}$, and finally $E = \{e_1, \ldots, e_L\}$ and $D = \{d_1, \ldots, d_L\}$ two sets of L trainable embeddings. Embeddings e_i and d_i serve as fixed identifiers for the latent variable z_i respectively in the encoder and in the decoder.

Given input token sequence w, the encoder $q_\phi(z|w) = \prod_l q_\phi(z_l|w)$ first yields parameters μ_l and σ_l to be used by the diagonal Gaussian distribution of each of the latent variables z_l as follows[4]:

[4]To simplify equations, we omit word embedding look-up tables and positional embeddings.

v	child	to wear	cloak	winter			
$k1$	nsubj	root	dobj	$\emptyset$	$\longrightarrow$	decoded (v, $k1$):	A child wears a cloak.
$k2$	agent	root	nsubjpass	pobj	$\longrightarrow$	decoded (v, $k2$):	A cloak is worn, in winter, by a child

Table 1: Example of interpretable values for the v and k in our model with $L = 4$. We display a sentence transiting from the active form to the passive form, to illustrate how different *keys* arranging the same *values* can lead to the same minimal semantic units being rearranged according to a different syntactic structure. We also stress that a different set of *keys* may omit or bring forth an element from the *values* vector (*e.g.* "winter" here above).

$$
\begin{aligned}
&\tilde{z} = \mathrm{Dec}(e; \mathrm{Enc}(w)) \\
&\forall\, l \quad \text{s.t. } 1 \le l \le L: \\
&\qquad \mu_l = M^{\mu}(\tilde{z}_l), \quad \sigma_l = \mathrm{SoftPlus}(M^{\sigma}(\tilde{z}_l)) \\
&\qquad z_l \sim \mathcal{N}(\mu_l; \sigma_l)
\end{aligned} \tag{3}
$$

Cross-Attention is also used by the ADVAE decoder to dispatch information from the *source* latent variable samples to the *target* generated sequence. Accordingly, using a beginning-of-sentence token w_0, $p_\theta(w|z) = \prod_i p_\theta(w_i|w_{<i}, z)$ yields probabilities for the categorical distribution of the generated tokens w by decoding latent variables z concatenated with their embeddings d:

$$
\begin{aligned}
&y = \mathrm{Cat}(d; z) \\
&\forall\, i \quad \text{s.t. } 1 \le i \le |w|: \\
&\qquad \tilde{w}_i = \overline{\mathrm{Dec}}(w_0, \ldots, w_{i-1}; \mathrm{Enc}(y)) \\
&\qquad w_i \sim \mathrm{Categorical}(\mathrm{Softmax}(M^{w}(\tilde{w}_i)))
\end{aligned}
$$

4 QKVAE: Using separate latent variables for Keys and Values

In this section, we describe the architecture of our model, the behavior it entails, and how we deal with the optimization challenges it poses.

4.1 QKVAE architecture

The modification we bring to ADVAE is aimed at controlling how information is selected from the latent space with the value of a newly introduced latent variable. We call this latent variable z^{syn}, and refer to the latent variables already formulated in ADVAE as $z^{sem} = \{z_1^{sem}, \ldots, z_L^{sem}\}$. z^{syn} is obtained with the same process as each z_l^{sem} (Eq. 3), *i.e.* by adding an additional identifier embedding e_s, and matrices $M^{\mu s}$ and $M^{\sigma s}$ to obtain its mean and standard-deviation parameters.

For the QKVAE Decoder, we modify the Transformer Decoder Dec into QKVDec so as to use Multi-Head Attention with separate inputs for keys and values instead of Cross-Attention :

$$
\mathrm{QKVDec}(T; S_K; S_V) = \tilde{T}_{D^{QKV}}, \text{ s.t. :}
$$

$$
\tilde{T}_d = \begin{cases} T \text{ if } d = 0, \text{ else:} \\ \mathrm{F}(\mathrm{MHA}(\mathrm{SA}(\tilde{T}_{d-1}), S_K, S_V) \end{cases}
$$

where D^{QKV} is the number of layers. Similar to $\overline{\mathrm{Dec}}$, we define $\overline{\mathrm{QKVDec}}$ to be the auto-regressive version of QKVDec. The QKVAE decoder yields probabilities for the generated tokens by using this operator on values given by z^{sem} concatenated with embeddings d, and keys given by a linear transformation on z^{syn}:

$$
\begin{aligned}
&v = \mathrm{Cat}(d; z^{sem}), \quad k = M^{s}(z^{syn}) \\
&\forall\, i \quad \text{s.t. } 1 \le i \le |w|: \\
&\qquad \tilde{w}_i = \overline{\mathrm{QKVDec}}(w_0, \ldots, w_{i-1}; k; v) \\
&\qquad w_i \sim \mathrm{Categorical}(\mathrm{Softmax}(M^{w}(\tilde{w}_i)))
\end{aligned}
$$

where M^s is a linear layer.[5] While ADVAE already uses Cross-Attention to encode and decode latent variables, our model uses separate variables to obtain keys and values for Multi-Head Attention in its decoder.

4.2 QKVAE Behavior

In the Multi-Head Attention of our decoder, z^{syn} controls keys, and z^{sem} controls values. In other words, the value of each z_l^{sem} is called to be passed to the target sequence according to its key which is given by the variable z^{syn}. Therefore, given a query, z^{syn} decides which content vector z_l^{sem} participates most to the value of the generated token at each generation step. To better get a gist of the kind of behavior *intended* by this construction, we assume in Table 1 for explanatory purposes, that our decoder has one layer and one attention head, that the value of each k^l in key matrices k_1 and k_2 corresponds to syntactic roles, and that each v^l informs on the realization of the corresponding syntactic role. Table 1 displays the resulting sentence when each of $k1$ and $k2$ are coupled with v.

[5]The output of M^s is reshaped to obtain a matrix of keys.

In the examples in Table 1, the generator uses a query at each generation step to pick a word in a manner that would comply with English syntax. Therefore, the key of each value should inform on its role in the target structure, which justifies syntactic roles as an adequate meaning for keys.

Although our model may stray from this possibility and formulate non-interpretable values and keys, keys will still inform on the *roles* of values in the target structure, and therefore influence the way values are injected into the target sequence. And given the fact that our model uses multiple layers and attention heads and the continuous nature of keys in Attention (as opposed to discrete syntactic role labels), our model performs a multi-step and continuous version of the behavior described in Table 1.

Injecting values into the structure of a sentence requires the decoder to model this structure. Previous works have shown that this is well within the capabilities of Transformers. Specifically, Hewitt and Manning (2019) showed that Transformers embed syntactic trees in their inner representations, Clark et al. (2019) showed that numerous attention heads attend to specific syntactic roles, and we (Felhi et al., 2021) showed that Transformer-based VAEs can capture the realizations of syntactic roles in latent variables obtained with Cross-Attention.

4.3 Balancing the Learning of z^{sem} and z^{syn}

Similar to ADVAE, we use a standard Normal distribution as a prior $p(z) = p(z^{sem})p(z^{syn})$ and train QKVAE with the β-VAE objective (Higgins et al., 2017) which is simply ELBo (Eq. 1) with a weight β on its Kullback-Leibler (KL) term. Higgins et al. (2017) show that a higher β leads to better unsupervised disentanglement. However, the KL term is responsible for a phenomenon called *posterior collapse* where the latent variables become uninformative and are not used by the decoder (Bowman et al., 2016). Therefore, higher values for β cause poorer reconstruction performance (Chen et al., 2018). To avoid posterior collapse, we follow Li et al. (2020a): *i)* We pretrain our model as an autoencoder by setting β to 0; *ii)* We linearly increase β to its final value (KL annealing; Bowman et al., 2016) and we threshold each dimension of the KL term with a factor λ (Free-Bits strategy; Kingma et al., 2016).

In preliminary experiments with our model, we observed that it tends to encode sentences using only z^{sem}. As we use conditionally independent posteriors[6] $q(z^{syn}|w)$ and $q(z^{sem}|w)$ for our latent variables, their KL terms (Eq. 1) can be written seperately, and they can therefore be weighted separately with different values of β. Using a lower β for z^{syn} as was done by (Chen et al., 2020) [7] did not prove effective in making it informative for the model. Alternatively, linearly annealing β for z^{sem} before z^{syn} did solve the issue. This intervention on the learning process was inspired by the work of Li et al. (2020c) which shows that latent variables used at different parts of a generative model should be learned at different paces.

5 Experiments

5.1 Setup

Data To compare our model to its supervised counterparts, we train it with data from the English machine-generated paraphrase pairs dataset ParaNMT (Wieting and Gimpel, 2018). More specifically, we use the 493K samples used by Chen et al. (2020)[8] to train their model VGVAE. Since our model is unsupervised, we only use the reference sentences (half the training set) to train our model. Using the development and test sets of ParaNMT, Chen et al. (2020) also provide a curated set of triplets formed by a target sentence (*target*), a semantic source (*sem_src*),and a syntactic source (*syn_src*). The semantic source is a paraphrase of the target sentence, while the syntactic source is selected by finding a sentence that is syntactically close to the target (*i.e.* edit distance between the sequence of PoS Tags of both sentences is low[9]) and semantically different from the paraphrase (has low BLEU score with it). Contrary to paraphrases in the training set of ParaNMT, paraphrases from this set were manually curated. These triplets are divided into a development set of 500 samples and a test set of 800 samples. We display results on the test set in the main body of the paper. The results on the development set, which lead to the same conclusions, are reported in Appendix A.

Training details & hyper-parameters Encoders and Decoders in QKVAE are initialized with pa-

[6] These posteriors are ADVAE encoders (Eq. 3).

[7] Although not explicitly mentioned in the paper, this is performed in their companion source code.

[8] https://drive.google.com/open?id=1HHDlUT_-WpedL6zNYpcN94cLwed_yyrP

[9] We follow Chen et al. (2020) by using this evaluation data, although edit distance between PoS tags might not be a good proxy for syntactic similarity.

rameters from BART (Lewis et al., 2020). After manual trial and error on the development set, we set the sizes of z^{syn} and z^{sem} to 768, and L to 4. Further Hyper-parameters are in Appendix B. We train 5 instances of our model and report the average scores throughout all experiments.

Baselines We compare our system to 4 previously published models, where 2 are supervised and 2 are unsupervised: *i) VGVAE (Chen et al., 2020):* a VAE-based paraphrase generation model with an LSTM architecture. This model is trained using paraphrase pairs and PoS Tags to separate syntax and semantics into two latent variables. This separation is used to separately specify semantics and syntax to the decoder in order to produce paraphrases; *ii) SynPG (Huang and Chang, 2021):* A paraphrase generation Seq2Seq model based on a Transformer architecture which also separately encodes syntax and semantics for the same purpose as VGVAE. This model is, however, trained using only source sentences with their syntactic parses, without paraphrases; *iii) Optimus (Li et al., 2020b):* A large-scale VAE based on a fusion between BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019) with competitive performance on various NLP benchmarks; *iv) ADVAE:* This model is QKVAE without its syntactic variable. The size of its latent variable is set to 1536 to equal the total size of latent variables in QKVAE.

Official open-source instances[10] of the 4 models above are available, which ensures accurate comparisons. The off-the-shelf instances of VGVAE and SynPG are trained on ParaNMT with GloVe[11] (Pennington et al., 2014) embeddings. We fine-tune a pre-trained Optimus on our training set following instructions from the authors. Similar to our model, we initialize ADVAE with parameters from BART(Lewis et al., 2020) and train 5 instances of it on ParaNMT with $L = 4$.

5.2 Syntax and Semantics Separation in the Embedding Space

We first test whether z^{syn} and z^{sem} respectively specialize in syntax and semantics. A syntactic (resp. semantic) embedding should place syntactically (resp. semantically) similar sentences close to each other in the embedding space.

	$z^{sem}\uparrow$	$z^{syn}\downarrow$
Supervised Models		
VGVAE	99.9	14.8
SynPG	93.4	26.5
Unsupervised Models		
Optimus	91.8	-
ADVAE	39.5	40.0
QKVAE	89.2	26.4

Table 2: The probability*100 that an embedding places a target sentence closer to its semantic source than it is to its syntactic source in the embedding space. Arrows (↑/↓) indicate whether higher or lower scores are better.

Using the (*target, sem_src, syn_src*) triplets, we calculate for each embedding the probability that *target* is closer to *sem_src* than it is to *syn_src* in the embedding space. For simplicity, we refer to the syntactic and semantic embeddings of all models as z^{syn} and z^{sem}. For Gaussian latent variables, we use the mean parameter as a representation (respectively the mean direction parameter from the von Mises-Fisher distribution of the semantic variable of VGVAE). We use an L2 distance for Gaussian variables and a cosine distance for the others. Since Optimus and ADVAE do not have separate embeddings for syntax and semantics *i)* We take the whole embedding for Optimus; *ii)*For ADVAE, we measure the above probability on the development set for each latent variable z_l (Eq. 3). Then, we choose the latent variable that places *target* sentences closest to their *sem_src* (resp. *syn_src*) as a semantic (resp. syntactic) variable. The results are presented in Table 2.

Table 2 clearly shows for QKVAE, SynPG, and VGVAE that the syntactic (resp. semantic) variables lean towards positioning sentences in the embedding space according to their syntax (resp. semantics). Surprisingly, the syntactic variable of our model specializes in syntax (*i.e.* has low score) as much as that of SynPG. The generalist latent variable of Optimus seems to position sentences in the latent space according to their semantics. Accordingly, we place its score in the z^{sem} column. Interestingly, the variables in ADVAE have very close scores and score well below 50, which shows that the entire ADVAE embedding leans more towards syntax. This means that, without the key/value distinction in the Attention-based decoder, the variables specialize more in structure than in content.

[10] VGVAE: github.com/mingdachen/syntactic-template-generation/; SynPG: github.com/uclanlp/synpg; Optimus: github.com/ChunyuanLI/Optimus; ADVAE: github.com/ghazi-f/ADVAE

[11] Gains could be observed with better embeddings for supervised models, but we stick to the original implementations.

	sem_src STED↑	*sem_src* TMA2↓	*sem_src* TMA3↓	*syn_src* STED↓	*syn_src* TMA2↑	*syn_src* TMA3↑	*target* STED↓	*target* TMA2↑	*target* TMA3↑
Control and Reference baselines									
sem_src	0.0	100	100	13.0	40.3	4.8	12.0	39.6	7.0
syn_src	13.0	40.3	4.8	0.0	100	100	5.9	84.3	45.8
Optimus	11.6	50.0	15.9	9.2	61.6	23.6	10.2	58.9	21.8
Supervised Models									
VGVAE	13.1	39.9	5.4	3.3	86.4	64.1	6.7	80.4	44.6
SynPG	11.7	41.9	18.0	13.5	74.1	10.5	13.1	69.1	13.3
Unsupervised Models									
ADVAE	11.9	47.3	14.0	10.3	54.3†	19.2†	11.1	52.3	17.0
QKVAE	12.7	40.2	7.8	7.2	68.2	39.5	8.9	63.9	28.1

Table 3: Syntactic transfer results. *STED* is the Syntactic Tree Edit Distance, and *TMA2/3* is the exact matching between constituency trees truncated at the 2^{nd}/3^{rd} level.

	sem_src M↑	*sem_src* PB↑	*syn_src* M↓	*syn_src* PB↓	*target* M↑	*target* PB↑
Control and Reference baselines						
sem_src	100	1.0	6.9	0.14	28.8	0.84
syn_src	6.9	0.14	100	1.0	12.1	0.16
Optimus	12.4	0.34	15.9	0.39	10.8	0.32
Supervised Models						
VGVAE	17.6	0.58	15.3	0.18	24.9	0.58
SynPG	45.9	0.87	8.0	0.13	25.2	0.75
Unsupervised Models						
ADVAE	8.0	0.19	8.3†	0.17	7.4	0.19
QKVAE	12.8	0.35	11.0	0.19	12.6	0.34

Table 4: Semantic transfer results. *M* is the Meteor score, and *PB* is the ParaBart cosine similarity.

5.3 Syntactic and Semantic Transfer

Similar to (Chen et al., 2020), we aim to produce sentences that take semantic content from *sem_src* sentences and syntax from *syn_src* sentences. For each of SynPG, VGVAE, and QKVAE we simply use the syntactic embedding of *syn_src*, and the semantic embedding of *sem_src* as inputs to the decoder to produce new sentences. Using the results of the specialization test in the previous experiment, we do the same for ADVAE by taking the 2 latent variables that lean most to semantics (resp. syntax) as semantic (resp. syntactic) variables. The output sentences are then scored in terms of syntactic and semantic similarity with *sem_src*, *syn_src* and *target*.

Control and reference baselines Beside model outputs, we also use our syntactic and semantic comparison metrics, explicited below, to compare *syn_src* and *sem_src* sentences to one another and to *target* sentences. Additionally, using Optimus, we embed *sem_src* and *syn_src*, take the dimension-wise average of both embeddings, and decode it. As VAEs are known to produce quality sentence interpolations (Bowman et al., 2016; Li et al., 2020b), the scores for this sentence help contrast a naïve fusion of features in the embedding space with a composition of well identified disentangled features.

Transfer metrics We measure the syntactic and semantic transfer from source sentences to output sentences. *i) Semantics:* For semantics, previous works (Chen et al., 2020; Huang and Chang, 2021) rely on lexical overlap measures such as BLEU (Papineni et al., 2001), ROUGE (Lin, 2004), and Meteor (Denkowski and Lavie, 2014). As will be shown in our results, the lexical overlap signal does not capture semantic transfer between sentences when this transfer is too weak to produce paraphrases. Therefore, we use Meteor (*M*) in conjunction with ParaBART (Huang et al., 2021) a model where BART (Lewis et al., 2020) is fine-tuned using syntactic information to produce neural representations that represent maximally semantics and minimally syntax. We measure the cosine similarity between sentences according to ParaBART embeddings (*PB*). *ii) Syntax:* We use the script of (Chen et al., 2020) to produce a syntactic tree edit distance (STED) between the constituency trees of sentences, as was done to assess VGVAE. Additionally, following the evaluation procedure designed by Huang and Chang (2021) for SynPG, we measure the Template Matching Accuracy between sentences, where the template is the constituency tree truncated at the second level (TMA2). TMA2 is the percentage of sentence pairs where such templates match exactly. We extend this measure by also providing it at the third level (TMA3). Results are presented in Tables 3 and 4. In both Tables, the comparison scores between sentences and *syn_src* that are not significantly[12] different from the same

[12]We consider differences to be significant if their associated t-test yields a p-value<0.01.

scores produced with regard to *sem_src* are marked with †.

Sanity checks with metrics and baselines We notice in Table 4 that using Meteor as a semantic similarity measure results in various inconsistencies. For instance, paraphrases *target* have a higher Meteor score with the syntactic sources than with interpolations from *Optimus*. It can also be seen that the Meteor score between outputs from VGVAE and both syntactic and semantic sources are rather close [13]. In contrast, ParaBART score behaves as expected across comparisons in Table 4. Consequently, we retain ParaBART score as a semantic similarity measure. In the following, we use the scores between *sem_src*, *syn_src*, and *target* (first two rows in Tables 4 and 3) as reference scores for unrelated sentences, paraphrase pairs, and syntactically similar sentences.

Comparing the supervised baselines VGVAE and SynPG greatly differ in scores. It can be seen that SynPG copies a lot of lexical items from its semantic input (high Meteor score) which allows for higher semantic similarity scores. However, Table 3 shows that SynPG transfers syntax from *syn_src* at a high level (high TMA2, but low TMA3). In contrast, VGVAE transfers syntax and semantics in a balanced way and achieves the best syntax transfer scores overall (lowest STED with *syn_src* and *target*).

Analysing the scores of QKVAE The semantic similarity scores *PB* of QKVAE outputs with *target* and *sem_src* are close to those of Optimus outputs. Although these scores are low compared to supervised models, they are notably higher than semantic similarity scores between unrelated sentences (*e.g.* *syn_src* and *sem_src*). However, in contrast to Optimus, QKVAE outputs display low PB scores with *syn_src*, which show that they draw very little semantic information from the syntactic sources. Concerning syntactic transfer in Table 3, QKVAE outputs share syntactic information with *syn_src* on all levels (low STED, and high TMA2 and TMA3). Our model is even competitive with SynPG on TMA2, and better on TMA3 and STED. As expected, the scores comparing QKVAE outputs to *sem_src* show that they share very little syntactic information. On the other hand, ADVAE shows poor transfer performance on syntax and semantics, with only slight differences between scores w.r.t *syn_src* and scores w.r.t *sem_src*.

[13] This was not observed by Chen et al. (2020), as they only compared outputs from VGVAE to the target paraphrases.

5.4 Comparing our Model to a Supervised Model with Less Data

Since VGVAE displays balanced syntactic and semantic transfer capabilities, we use it for this experiment where we train it on subsets of sizes in $\{10K, 25K, 50K, 100K\}$ from its original training data. Our goal is to find out how much labeled data is needed for VGVAE to outperform our unsupervised model on both transfer metrics.

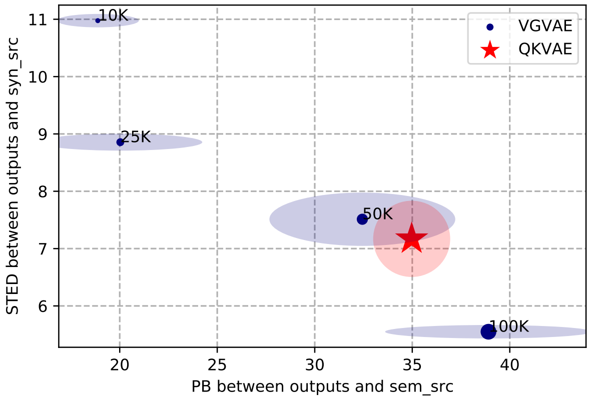

Figure 1: Plotting STED w.r.t *syn_ref* and the PB cosine similarity w.r.t *sem_ref* for VGVAE with different amounts of labeled data and for QKVAE. Points are scaled proportionally to the amount of training data. The vertical and horizontal diameters of each ellipse are equal to the standard deviation of the associated data points and axes.

In Figure 1, we plot for QKVAE and instances of VGVAE the *STED* of their outputs w.r.t *syn_src* and the *PB* of these outputs w.r.t *sem_src*. All values are averages over 5 runs, with standard deviations plotted as ellipses. Figure 1 shows that to outperform QKVAE on syntactic and semantic transfer, VGVAE needs more than 50K labeled samples.

6 Discussion and conclusion

In Table 5, we display example outputs of SynPG, VGVAE, and QKVAE along with their syntactic sources, semantic sources, and targets. We generally observed that the outputs of QKVAE range from paraphrases (line 6) to broadly related sentences (line 3). As was shown by our quantitative results, outputs from VAE-based models (VGVAE and QKVAE) share relatively few lexical items with the semantic input. This can be seen in the qualitative examples where they often swap words in the semantic source with closely related words (*e.g.*

sem_src	*syn_src*	SynPG	VGVAE	QKVAE	*target*
we have destroyed the 49th armored division.	concomitant usage is not recommended.	we have destroyed the 49th armored division.	armored division hasn't destroyed.	this military force will be destroyed.	49th armored division has been destroyed .
let the fire burn and put a piece of hot iron in it.	sing a song. sing a song for boys.	don't put the fire in it burn a hot piece of iron and fire.	burn the fire. put the iron on burns.	come on fire. get a fire on it.	keep this fire going. keep a piece of hot iron on it.
they took the lunch boxes ?	have you given me your hands ?	do they boxes took the lunch ?	have they taken them your snacks ?	have you heard of some lunch ?	have they taken the lunch boxes ?
does it have a coach ?	that's a phone switcher, right ?	how does it have a coach ?	that's a coach coach, right ?	that's a warden, huh?	it has a coach, no ?
an old lady in a cemetery.	that is a bad time for a war.	there's a lady in an old cemetery.	that's an old lady in the cemetery.	this is a strange place for a woman.	there is an old lady in the cemetery.
don't be afraid.	there are still many places to go.	you don't be afraid.	there aren't be afraid to be.	there will be no need to worry.	there is no need to be afraid .
isn't there a door open ?	the machines are still good, right ?	a isn't open door there ?	the doors aren't open, right ?	the door will be open, okay?	there is a door open, right ?

Table 5: Syntactic sources (*syn_src*), semantic sources (*sem_src*), the sentences produced when using them with different models, and the corresponding correct paraphrases (*target*).

"armored division" to "military force" in line 1, or "lunch boxes" to "snacks" in line 2). We attribute this quality to the smoothness of the latent space of VAEs which places coherent alternative lexical choices in the same vicinity. The examples above also show that our model is capable of capturing and transferring various syntactic characteristics such as the passive form (line 1), the presence of subject-verb inversion (lines 3, 4, and 7), or interjections (lines 4 and 6).

We presented QKVAE, an unsupervised model which disentangles syntax from semantics without syntactic or semantic information. Our experiments show that its latent variables effectively position sentences in the latent space according to these attributes. Additionally, we show that QKVAE displays clear signs of disentanglement in transfer experiments. Although the semantic transfer is moderate, syntactic transfer with QKVAE is competitive with SynPG, one of its supervised counterparts. Finally, we show that VGVAE, a supervised model, needs more than 50K samples to outperform QKVAE on both syntactic and semantic transfer.

We plan to extend this work in three directions: i) Finding ways to bias representations of each z_l^{sem} towards understandable concepts; *ii)* Applying QKVAE to non-textual data since it is data agnostic (e.g. to rearrange elements of a visual landscape.); *iii)* Investigating the behavior of QKVAE on other languages.

Acknowledgments

This work is supported by the PARSITI project grant (ANR-16-CE33-0021) given by the French National Research Agency (ANR), the *Laboratoire d'excellence "Empirical Foundations of Linguistics"* (ANR-10-LABX-0083), as well as the ONTORULE project. It was also granted access to the HPC resources of IDRIS under the allocation 20XX-AD011012112 made by GENCI.

References

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xinyu Dai, and Jiajun Chen. 2019. Generating Sentences from Disentangled Syntactic and Semantic Spaces. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 6008–6019.

Melika Behjati and James Henderson. 2021. Inducing Meaningful Units from Character Sequences with Slot Attention. *Arxiv*.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings*, pages 10–21.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. A multi-task approach for disentangling syntax and semantics in sentence representations. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:2453–2464.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2020. Controllable paraphrase generation with a syntactic exemplar. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 1, pages 5972–5984.

Tian Qi Chen, Xuechen Li, Roger Grosse, and David Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. In *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*.

Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. 2020. Improving Disentangled Text Representation Learning with Information-Theoretic Guidance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7530–7541.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *BlackBoxNLP@ACL*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:4171–4186.

Wenyu Du, Zhouhan Lin, Yikang Shen, Timothy J O'Donnell, Yoshua Bengio, and Yue Zhang. 2020. Exploiting Syntactic Structure for Better Language Modeling: A Syntactic Distance Approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 6611–6628.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars. *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, pages 199–209.

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R. Arabnia. 2021. A Brief Review of Domain Adaptation. In *Advances in Data Science and Information Engineering*, pages 877–894.

Ghazi Felhi, Joseph Le Roux, and Djamé Seddah. 2021. Towards Unsupervised Content Disentanglement in Sentence Representations via Syntactic Roles. In *1st CtrlGen: Controllable Generative Modeling in Language and Vision Workshop at NeurIPS 2021*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:4129–4138.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. B-VAE: Learning basic visual concepts with a constrained variational framework. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–22.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger P Levy. 2020. A Systematic assessment of syntactic generalization in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1725–1744.

James Y. Huang, Kuan-Hao Huang, and Kai-Wei Chang. 2021. Disentangling Semantics and Syntax in Sentence Embeddings with Pre-trained Language Models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1372–1379, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kuan-hao Huang and Kai-wei Chang. 2021. Generating Syntactically Controlled Paraphrases without Using Annotated Parallel Pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Stroudsburg, PA, USA. Association for Computational Linguistics.

Drew A. Hudson and Christopher D. Manning. 2018. Compositional Attention Networks for Machine Reasoning. *International Conference on Learning Representations*, 333(6045):975–8.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 11–20.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2020. Disentangled representation learning for non-parallel text style transfer. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 424–434.

Yoon Kim, Alexander M Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019. Unsupervised recurrent neural network grammars. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:1105–1117.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, Ml, pages 1–14.

Jordan Kodner and Nitish Gupta. 2020. Overestimation of syntactic representation in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 1757–1762.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 4077–4091.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2020a. A surprisingly effective fix for deep latent variable modeling of text. In *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 3603–3614.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020b. Optimus: Organizing Sentences via Pre-trained Modeling of a Latent Space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dingcheng Li, Hongliang Fei, Shaogang Ren, and Ping Li. 2021. A deep decomposable model for disentangling syntax and semantics in sentence representation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4300–4310, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiyuan Li, Jaideep Vitthal Murkute, Prashnna Kumar Gyawali, and Linwei Wang. 2020c. Progressive Learning and Disentanglement of Hierarchical Representations. *International Conference on Learning Representations*.

Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004, Barcelona, Spain*, pages 74–81.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. 2020. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 2020-Decem(NeurIPS):1–27.

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. 2019. Learning under Concept Drift: A Review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363.

Rebecca Marvin and Tal Linzen. 2020. Targeted syntactic evaluation of language models. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 1192–1202.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, volume 371, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-Theoretic Probing for Linguistic Structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Shauli Ravfogel, Yanai Elazar, Jacob Goldberger, and Yoav Goldberg. 2020. Unsupervised distillation of syntactic information from contextualized word representations. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 91–106, Online. Association for Computational Linguistics.

Danilo Jimenez Rezende, S. M. Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. 2016. Unsupervised learning of 3D structure from images. In *Advances in Neural Information Processing Systems*, Nips, pages 5003–5011.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. 2022. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–80.

Djamé Seddah, Farah Essaidi, Amal Fethi, Matthieu Futeral, Benjamin Muller, Pedro Javier Ortiz Suárez, Benoît Sagot, and Abhishek Srivastava. 2020. Building a User-Generated Content North-African Arabizi Treebank: Tackling Hell. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1139–1150. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *35th International Conference on Machine Learning, ICML 2018*, 10:7322–7330.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *7th International Conference on Learning Representations, ICLR 2019*, pages 1–14.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 155(1-2):105–145.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Andros Tjandra, Ruoming Pang, Yu Zhang, and Shigeki Karita. 2021. Unsupervised learning of disentangled speech content and style representation. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 4:3191–3195.

Jannis Vamvas and Rico Sennrich. 2021. On the Limits of Minimal Pairs in Contrastive Evaluation. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 58–68, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *NeurIPS*, Nips.

Sarah Wiegreffe and Yuval Pinter. 2020. Attention is not not explanation. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 11–20.

John Wieting and Kevin Gimpel. 2018. ParanMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:451–462.

Peng Xu, Jackie Chi Kit Cheung, and Yanshuai Cao. 2020. On Variational Learning of Controllable Representations for Text without Supervision. *The 37th International Conference on Machine Learning (ICML 2020)*.

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. 2019. Syntax-Infused Variational Autoencoder for Text Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2069–2078, Stroudsburg, PA, USA. Association for Computational Linguistics.

A Results on the development set

We hereby display the scores on the development set. The encoder scores concerning the specialization of latent variables are in Table 6, while the transfer scores are in Table 7 for semantics, and Table 8 for syntax. The values on the development set concerning the comparison of QKVAE with VGVAE trained on various amounts of data is in Figure 2.

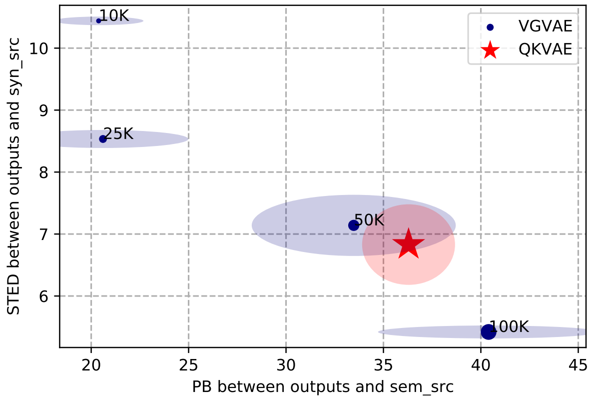

Figure 2: Plotting STED w.r.t *syn_ref* and the PB cosine similarity w.r.t *sem_ref* for VGVAE with different amounts of labeled data and for QKVAE. Points are scaled proportionally to the amount of training data. The vertical and horizontal diameters of each ellipse are equal to the standard deviation of the associated data points and axes.

B Hyper-parameters

Hyper-parameter values The β weight on the KL divergence is set to 0.6 for z^c and to 0.3 for z^s, and the λ threshold for the Free-Bits strategy is set to 0.05. KL annealing is performed between steps 3K and 6K for z^{sem}, and between steps 7K and 20K for z^{syn}. The model is trained using Adafactor (Shazeer and Stern, 2018), a memory-efficient version of Adam (Kingma and Ba, 2015). Using a batch size of 64, we train for 40 epochs, which takes about 30 hours on a single Nvidia GEForce RTX 2080 GPU. We use 4 layers for both Transformer encoders and decoders. The encoders (resp. decoders) are initialized with parameters from the 4 first layers (resp. 4 last layers) of BART encoders (resp. decoders). In total, our model uses 236M parameters.

	$z^{sem}\uparrow$	$z^{syn}\downarrow$
Supervised Models		
VGVAE	99.0	16.4
SynPG	91.6	31.2
Unsupervised Models		
Optimus	89.4	-
ADVAE	41.0	40.3
QKVAE	86.7	27.0

Table 6: The probability*100 that an embedding places a target sentence closer to its semantic source than it is to its syntactic source in the embedding space. (development set results)

	sem_src M↑	*sem_src* PB↑	*syn_src* M↓	*syn_src* PB↓	*target* M↑	*target* PB↑
Control and Reference baselines						
sem_src	100	1.0	7.4	0.13	27.4	0.82
syn_src	7.4	0.13	100	1.0	12.0	0.16
Optimus	13.00	0.35	13.4	0.34†	10.5	0.32
Supervised Models						
VGVAE	18.3	0.58	15.2	0.17	23.0	0.57
SynPG	47.6	0.86	7.8	0.11	24.4	0.73
Unsupervised Models						
ADVAE	9.0	0.20	8.1	0.17	7.7	0.19
QKVAE	13.4	0.36	11.3	0.19	12.9	0.35

Table 7: Semantic transfer results (development set results)

Manual Hyper-parameter search Given that the architecture for Transformer layers is fixed by BART, we mainly explored 3 parameters: number of latent variables L, number of Transformer layers, values for β. Our first experiments have shown that setting L to 8 or 16 does not yield good re-

	sem_src STED↑	*sem_src* TMA2↓	*sem_src* TMA3↓	*syn_src* STED↓	*syn_src* TMA2↑	*syn_src* TMA3↑	*target* STED↓	*target* TMA2↑	*target* TMA3↑
Control/Ceiling baselines									
sem_src	0.0	100	100	11.9	46.4	6.8	10.9	47.0	7.3
syn_src	11.9	46.4	6.8	0.0	100	100	6.0	81.6	45.0
Optimus	9.7	58.2	20.6	9.2†	61.6†	22.6†	9.9	59.6	18.4
Supervised Models									
VGVAE	11.9	45.4	6.8	3.2	84.2	58.2	6.7	77.6	39.0
SynPG	9.3	49.4	21.4	12.2	73.0	12.2	12.2	68.6	13.0
Unsupervised Models									
ADVAE	10.1	53.4	18.6	9.8†	55.0†	17.4†	10.5	52.8	15.4
QKVAE	11.4	45.0	9.1	6.8	66.4	37.4	8.6	63.0	26.9

Table 8: Syntactic transfer results (development set results)

sults, which is probably due to the fact that a high L raises the search space for possible arrangements of values with keys, and consequently makes convergence harder. Concerning the number of layers, we observed that results with the full BART model (6 layers) have high variance over different runs. Reducing the number of layers to 4 solved this issue. In regards to β, we observed that it must be 0.6 or less for the model to produce adequate reconstructions and that it is beneficial to set it slightly lower for z^{syn} than for z^{sem} so as to absorb more syntactic information with z^{syn}.