DISCRETE TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

The transformer has become a central model for many NLP tasks from translation to language modeling to representation learning. Its success demonstrates the effectiveness of stacked attention as a replacement for recurrence for many tasks. In theory attention also offers more insights into the model's internal decisions; however, in practice when stacked it quickly becomes nearly as fully-connected as recurrent models. In this work, we propose an alternative transformer architecture, discrete transformer, with the goal of better separating out internal model decisions. The model uses hard attention to ensure that each step only depends on a fixed context. Additionally, the model uses a separate "syntactic" controller to separate out network structure from decision making. Finally we show that this approach can be further sparsified with direct regularization. Empirically, this approach is able to maintain the same level of performance on several datasets, while discretizing reasoning decisions over the data.

1 INTRODUCTION

The transformer has achieved state-of-the-art performances in a variety of sequence modeling tasks, including language modeling (Radford et al., 2019), machine translation (Vaswani et al., 2017), question answering (Radford et al., 2018; Devlin et al., 2018), among others. To facilitate parallel training, as well as to reduce the path length of the dependencies, transformer dispenses recurrence and builds up hidden states by attending to the source side (inter-attention) and attending to its past predictions (self-attention) with multiple heads in multiple layers (Vaswani et al., 2017).

Compared to recurrent models the attention mechanism adds some "interpretability" to a model's decision (Bahdanau et al., 2014; Xu et al., 2015; Chan et al., 2015). However, in the commonly used soft attention mechanism (Luong et al., 2015) each input element receives non-zero weight, and so it is unclear whether the magnitude of attention weights reflects the relative importance of the corresponding inputs (Jain & Wallace, 2019). To make things worse, due to the existence of multiple stacked attention layers in transformer, it becomes even harder to discriminate the contributions of each input to the final decisions made by the model.

Can we force the transformer to make sharper, discrete internal decisions? In this work, we consider a variant of the transformer architecture with the goal of maintaining performance while forcing discrete decisions. Specifically, we consider a *discrete transformer* with three changes to the architecture: (a) we propose to treat attention as a categorical latent variable (Deng et al., 2018; Shankar et al., 2018) and use hard attention mechanism to get discrete attention decisions (Xu et al., 2015), (b) we propose to separate out the querying mechanism from value computation into intertwine soft "syntactic" and hard "semantic" model streams, and (c) we consider extension to the discrete transformer to allow for further additions such as attention sparsity regularization.

Training of the model is very similar to standard transformer training. The key benefits come at inference time. First, we can use a simple decoding procedure where we take argmax attentions such that each intermediate representation is only built up based on the subset of the attended lower layer outputs. In turn, each final prediction uses limited receptive field, and we can even the guarantee that any hidden state does not depend on input elements not being directly attended to. Second, we can split out attention prediction from computation, and even fix the structure of the feed-forward network for a given example.

To validate this approach, we perform experiments on several tasks. We first validate that with proper attention and sparsity regularization the model can learn the truly necessary attentions on a synthetic

language modeling task. Next on two real world machine translation datasets, we show that with our approach we can learn transformer models using limited context for making predictions while not deteriorating their performance by too much, indirectly validating the selectiveness of the attention mechanism.

The rest of the paper is organized as follows: In Section 2 we draw the connections of our work to the literature. We introduce background and discuss our approach in Sections 3, 4 and 5. Experiments, results and analyses are presented in Sections 6 and 7, and we conclude our paper in Section 8.

2 Related Work

Attention has been used to imply transparency into model's prediction. This is crucially important in domains health care (Caruana et al., 2015; Choi et al., 2016; Rajkomar et al., 2018) but has also been used in other natural language sequence modeling tasks (Rush et al., 2015; Deng et al., 2017; Alvarez-Melis & Jaakkola, 2017). Since the soft attention mechanism assigns non-zero weights everywhere, to get interpretability, a general assumption is that larger attention weights correspond to higher importance in making a decision (Unanue et al., 2018). However, a recent work (Jain & Wallace, 2019) shows that attention magnitude does not correlate well with gradient-based measures of input elements importance (Selvaraju et al., 2017).

To get around with the difficulty of credit assignment in soft attention, researchers have proposed to use sparse attentions. Peters et al. (2018) uses sparsemax (Martins & Astudillo, 2016) to induce sparse attention structures. Lei et al. (2016) model attention as Bernoulli random variables and use an encoder to produce the final prediction only from the attended input elements such that the final predictions can be rationalized. To optimize the final objective, Lei et al. (2016) apply policy gradients. Our work follows the spirit of their work, but we consider multi-head multi-layer attentions in transformer, which subjects REINFORCE algorithm to large gradient variance. Instead we use the Gumbel-Softmax trick (Jang et al., 2016; Maddison et al., 2016) to get reparameterizable samples and reduce the gradient variance (Kingma & Welling, 2013).

Broadly speaking, there are two directions of work towards improving interpretability: model interpretability and prediction interpretability (Alvarez-Melis & Jaakkola, 2017). Prediction interpretability relies on an external interpreter that is both interpretable and locally consistent with the black box model being explained, through which we can analyze the causal relationships between inputs and outputs (Ribeiro et al., 2016). In model interpretability, researchers try to build models that are commonly regarded as interpretable (Louppe, 2014; Calders et al., 2013; Doshi-Velez & Kim, 2017; Peters et al., 2018). Our approach aims to improve model interpretability by modifying the attention mechanism and objective function where we implicitly assume that more sparsity in the attention structure implies more interpretability, rather than relying on another model to analyze an existing one. While our approach does not directly lead to prediction interpretability, we can draw connections between our approach and the prediction interpretability framework of Alvarez-Melis & Jaakkola (2017) if we consider local permutations of input embeddings: for inputs not being directly or indirectly attended to at a specific prediction step, the local interpreter does not need to use them at all, hence there is no causal relationship between these inputs and the prediction.

The separation between query mechanism and value computation resembles the two-stream attention mechanism in XLNet (Yang et al., 2019), where a separate query stream is introduced in addition to the normal content stream to enable the usage of target position information while avoiding "cheating" to work with arbitrary generation factorization order. Recently, Russin et al. (2019) used word embeddings as content vectors whereas the attention is computed based on the outputs of an LSTM network. This approach shares a similar goal with ours to separate syntax and semantics, but transformer presents its unique challenges due to the existence of multiple layers and multi-headed attentions and its lack of recurrence.

3 Background: Transformer

We begin by briefly reviewing the transformer architecture that will serve as the basis for this work. Specifically we will consider a transformer for simplified classification task over a sentence (experiments will expand this to autoregressive models). Let $x_{1:T}$ be a sequence of input tokens and y is a

discrete output label. We begin by encoding tokens with a position-specific embedding function e to vectors $\boldsymbol{h}_i^0$. Each layer of the transformer then produces new vectors under the following recursion[1]:

$$\begin{aligned}
\boldsymbol{B}_i &\leftarrow \text{FFN}(\boldsymbol{h}_i^{(l-1)}) \ \forall \ i \in 1, \ldots, T \\
\boldsymbol{K}, \boldsymbol{V}, \boldsymbol{Q} &\leftarrow (\boldsymbol{B}\boldsymbol{W}^{(K)}, \boldsymbol{B}\boldsymbol{W}^{(V)}, \boldsymbol{B}\boldsymbol{W}^{(Q)}) \\
\boldsymbol{A} &\leftarrow \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}) \\
\boldsymbol{h}_i^{(l)} &\leftarrow \boldsymbol{h}_i^{(l-1)} + \boldsymbol{V}\boldsymbol{A}_i \ \forall \ i \in 1, \ldots, T
\end{aligned}$$

Here l is the layer index, FFN is a large, feed-forward NN, $\boldsymbol{W}$ are learned projection parameters, and d_k is a constant for the size of the network. The key intermediary terms are the key $\boldsymbol{K}$, value $\boldsymbol{V}$, and query $\boldsymbol{Q}$ matrices which contain a vector for each position $1 \ldots T$, and the attention matrix $\boldsymbol{A}$ which gives a distribution for each position over its "attention" to all other positions. This attention is computed from the queries and keys and then used to take a convex combination of the values. At the final layer L we can then make a prediction utilizing $\boldsymbol{h}^{(L)}$, e.g. a softmax over all possible y labels.

The main processing work of the transformer happens in the FFN layers which contain the majority of the non-embedding parameters. These can be thought of as large, width-1 convolutional networks that process the full sentence at each layer. However, for these layers to be effective it is crucial that information from other tokens be aggregated together. The attention layer serves as the single source of this aggregation in the model. Attention is the only point where inter-word information routing occurs.

Because attention is central for routing and determines the receptive field of the transformer, it has been a main source of study for the transformer and related models. If attention can be understood then in theory the interconnection between words can be mapped and perhaps even manipulated.

Unfortunately much of this work has so far produced negative results. The underlying problem is that while any one attention layer may target a small amount of keys, in aggregate repeated applications of multi-headed attention quickly connect every position to every other. Learned attention acts in a soft way and roughly pools together all elements into a vector. While this may be usable for high-level analysis, it does not allow us to truly separate out anything about the model decisions.

4 Discrete Attention Transformer

In several recent works (Deng et al., 2018; Shankar et al., 2018), researchers have explored alternatives to soft attention for single-layer (pre-transfomer) attention models. The goal of this work has been to learn models that can replace soft-attention with latent control variables that select a single position to use. This form of discrete attention can produce models that make these intermediate decisions explicit, which has been shown to produce models that perform as well or better than soft models.

We begin by considering applying this approach directly to transformer. Formally, we can replace the above deterministic equations with an intermediate sampling step:

$$\begin{aligned}
\boldsymbol{B}_i &\leftarrow \text{FFN}(\boldsymbol{h}_i^{(l-1)}) \ \forall \ i \in 1, \ldots, T \\
\boldsymbol{K}, \boldsymbol{V}, \boldsymbol{Q} &\leftarrow (\boldsymbol{B}\boldsymbol{W}^{(K)}, \boldsymbol{B}\boldsymbol{W}^{(V)}, \boldsymbol{B}\boldsymbol{W}^{(Q)}) \\
\boldsymbol{A} &\leftarrow \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}) \\
\boldsymbol{h}_i^{(l)} &\leftarrow \boldsymbol{h}_i^{(l-1)} + \boldsymbol{V}_{z_i^l} \text{ where } z_i^l \sim \text{Cat}(\boldsymbol{A}_i) \ \forall \ i \in 1, \ldots, T
\end{aligned}$$

[1]We elide layernorm, dropout, most residual connections, and multi-headedness for presentational simplicity. These are all included in our final model.

This makes the transformer stochastic and requires computing, $p(y|x) = \sum_z p(y, z|x)$ since we do not observe z. In a single layer model such as a sequence-to-sequence RNN, this might be a tractable sum, but for stacked attention, we would need to be able to sum out over all possible choices for z. In transformer this is combinatorial, and computing this term even for single-headed attention is $O(T^L)$.

Because of this complexity several alternative methods have been proposed in the literature. Given the success of soft attention for transformer, we opt for using a Gumbel-Softmax approach for training, which modulates between soft training and hard inference (Jang et al., 2016; Maddison et al., 2016).

The Gumbel-Softmax approach gives a continuous approximation to sampling from the categorical distribution. Given a categorical distribution defined by log probabilities $\boldsymbol{l}$, the Gumbel-Softmax generative process is defined by first sampling $U_k \sim \text{Uniform}(0, 1)$, and then returns

$$\text{Gumbel-Softmax}(\boldsymbol{l}) \propto \exp((l_k + g_k)/\tau)$$

where $g_k = \log(\log(U_k))$ is called Gumbel noise (the distribution of g_k is Gumbel distribution), and τ is a temperature parameter controlling the entropy of the distribution. As $\tau \to 0$, samples given by the Gumbel-Softmax function conforms to the same distribution as one-hot categorical samples.

At training time, we can use this approach to obtain differentiable samples approximating sampling from the categorical attention distributions. Unlike REINFORCE, we can apply reparameterization to get a low variance gradient estimator and directly back-propagate through it in modern deep learning libraries. We use fixed temperature τ throughout training but we tune its value on the validation set.

At test time, we replace Gumbel-Softmax with the argmax from the vector. This corresponds to a greedy choice over the random variable. This test time model is almost identical to the original transformer: we simply replace softmax with argmax. The main benefit of this method though is that for every position we have a fixed tree of all the previous words that influenced it. We can effectively guarantee that if a word was not in this tree, then it did not contribute to the hidden state value of that position.

5 Transformer with Separate Syntactic and Semantic Streams

By utilizing discrete attention, we can ensure that model routing is done through hard choices by the transformer. Since the final computation is done only based on the hidden state at the top layer, we can ensure that this decision was only made based on the hard pathway to the original words.

To formalize this concept we consider the continuous receptive field of any hidden state $\boldsymbol{h}_i^{(l)}$, that is the input vectors that directly determined its current value. The structure of hard attention ensures that the receptive field is defined by the recursion $r(i, l) = r(i, l-1) \cup r(z_i^{(l)}, l-1)$ where $r(i, 0) = \{i\}$ and z_i^l is the hard sample taken at layer l for position i. While this receptive field grows exponentially with layers, its branching factor is much more constrained than with soft attention.

However, we note that this same property can be obtained by giving more flexibility to the calculation of keys and values used in hard attention. In fact, these calculations can be kept soft throughout the entirety of the transformer without changing the continuous receptive field.

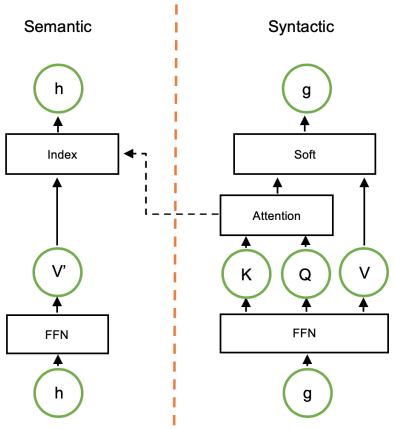


Figure 1: Discrete Transformer architecture. Syntactic transformer stream computes the attention distribution which is used to produce next hidden states while also constructing the semantic architecture through latent hard attention.

We therefore propose an extension to the model structure of transformer that aims to separate out the "synatactic" routing control of the model from the "semantic" computation part. Motivated by Russin et al. (2019), we follow the distinction that the semantic part of the model should consist of a fixed, sparse feed forward network, whereas the syntactic part is free to consider the entire sentence at any step. We build these together using a two stream transformer network.

To build this model, we make the observation that values should be computed only by the semantic network, and that keys and queries should be computed only by the syntactic network. Let the syntatic representation at timestep i in layer l be $\boldsymbol{g}_i^{(l)}$ and the semantic representation be $\boldsymbol{h}_i^{(l)}$. The first layer representations are set to the corresponding word embedding, $\boldsymbol{g}_i^{(0)} = e_g(x_i)$ and $\boldsymbol{h}_i^{(0)} = e_h(x_i)$.

For each attention layer l, the two streams are updated as follows:

$$
\begin{array}{rcl}
\text{Semantic Stream} & & \text{Syntactic Stream} \\
\boldsymbol{C}_i \leftarrow \text{FFN}(\boldsymbol{h}_i^{(l-1)}) & & \boldsymbol{B}_i \leftarrow \text{FFN}(\boldsymbol{g}_i^{(l-1)}) \ \forall \ i \in 1, \ldots, T \\
\boldsymbol{V}^{'} \leftarrow \boldsymbol{C}\boldsymbol{W}^{(h)} & & \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{Q} \leftarrow (\boldsymbol{B}\boldsymbol{W}^{(K)}, \boldsymbol{B}\boldsymbol{W}^{(V)}, \boldsymbol{B}\boldsymbol{W}^{(Q)}) \\
\multicolumn{3}{c}{\boldsymbol{A} \leftarrow \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}})} \\
\boldsymbol{h}_i^{(l)} \leftarrow \boldsymbol{h}_i^{(l-1)} + \boldsymbol{V}'_{z_i^l} \text{ where } z_i^l \sim \text{Cat}(\boldsymbol{A}_i) & & \boldsymbol{g}_i^{(l)} \leftarrow \boldsymbol{g}_i^{(l-1)} + \boldsymbol{V}\boldsymbol{A}_i \ \forall \ i \in 1, \ldots, T
\end{array}
$$

Note that the syntactic stream (right) is completely independent of the semantic stream and could even be computed before hand. Furthermore assuming a fixed syntactic distribution the semantic distribution becomes a vanilla feedforward network. Practically, the two streams distributions can be trained together as a single network. Each attention step can be aligned as well as the FFN computations.

5.1 Extension: Sparsifying the Receptive Field

A secondary benefit of utilizing an explicit latent variable within the model is the ability to impose structural constraints on the variable directly based on prior knowledge such as structured attention (Kim et al., 2017) or sparse attention (Niculae & Blondel, 2017). Here we consider a way to penalize the size of the receptive field at the final layer $|\bigcup_i r(i, L)|$. We want a differentiable version of $|\bigcup_i r(i, L)|$. Let's use s_{ij}^l to denote whether the representation of token i relies on embedding of token j at layer l, i.e. $j \in r(i, l)$, then we have the recursion that

$$
s_{ij}^l = \min \left(s_{ij}^{l-1} + \sum_k z_{ik}^l s_{kj}^{l-1}, 1 \right) \tag{1}
$$

Where the internal representation of token i depends on embedding of token j if $s_{ij}^{l-1} = 1$ (due to residual connections, the dependencies of a layer below are also the current dependencies) or if i attends to k at layer $l-1$ and $s_{kj}^{l-1} = 1$ (the dependencies of the token attended to also become the current dependencies). Based on this recursion (and initial conditions that $s_{ij}^0 = \mathbb{1}(i = j)$), the final layer receptive field size can be calculated as $|\bigcup_i r(i, L)| = \sum_i \min \left(\sum_j s_{ij}^L, 1 \right)$. We note that since during training z comes from the Gumbel-Softmax, the z_{ik} values are computed as a soft approximation (as opposed to indices). Therefore it provides useful gradients and can be directly applied as a regularizer.

6 Experimental Setup

We run experiments on several different benchmark datasets including machine translation and language modeling. We also test if our approach is able to recover the true underlying dependencies

with the sparsity regularizer. For each dataset, we start with a strong transformer baseline, with the goal of inducing a similarly performing model with explicit latent structure.

Data For translation, we conduct experiments on two machine translation datasets: IWSLT (Cettolo et al., 2014), a standard small-scale benchmark, and the much larger WMT English to German (Bojar et al., 2017). To test whether our approach is possible to discover the true dependencies in the data, we also constructed a synthetic language modeling task with known underlying dependencies.

Architecture and Hyperparameters While we describe the method using a simplified model, for experiments, we model our system directly off standard transformer models. For translation that means using both an encoder and a decoder transformer stack. Our DISCRETE TRANSFORMER uses encoder and decoder stacks analogous to those in TRANSFORMER. For self attention, we replace the scaled dot-product attention in TRANSFORMER by our two stream attention. The input consists of queries $\boldsymbol{Q}$ and keys $\boldsymbol{K}$ obtained from the syntactic representation g and two values $\boldsymbol{V}$ and $\boldsymbol{V}^{'}$, one each from the syntactic representation g and the semantic representation h respectively. Multi-Head Attention is computed analogously.

For encoder-decoder context attention, the syntactic stream of the encoder provides the keys $\boldsymbol{K}$ and the syntactic values $\boldsymbol{V}$, the encoder's semantic stream provides the semantic values $\boldsymbol{V}^{'}$, and the decoder's syntactic stream provides the queries $\boldsymbol{Q}$.

For WMT, we use the base model with $d_{\text{model}} = 512$, $d_{\text{ff}} = 2048$, $L = 6$, $h = 8$ (we refer to Vaswani et al. (2017) for hyperparameter definitions). For IWSLT, we use $d_{\text{model}} = 512$, $d_{\text{ff}} = 1024$, $L = 6$, $h = 4$ since it is more prone to overfitting. We implement our models based on Fairseq (Ott et al., 2019). For WMT a single model was obtained by averaging the last 5 checkpoints saved every 1000 update steps. The Gumbel temperature is set to 1 throughout this paper, and we set the sparisity regularizer strength to 0.1 for the synthetic language modeling task.

For language modeling, we we use only the decoder network of the transformer for language modeling. We use $d_{\text{model}} = 64$, $d_{\text{ff}} = 256$, $L = 4$, $h = 2$ for the synthetic stack language modeling task.

Baselines The first baseline we consider is the normal transformer model with soft attention TRANSFORMER. Then we make the attentions discrete by applying Gumbel-Softmax at training time and argmax at test time, which we term SINGLE STREAM DISCRETE TRANSFORMER. We compare those baselines to our model equipped with discrete attention and two-stream attention DISCRETE TRANSFORMER. We also consider experiments separating out the syntactic and semantic streams.

7 RESULTS AND ANALYSIS

7.1 PRELIMINARY ANALYSIS: STACK LANGUAGE

To test whether the sparse attentions learned by our approach correspond to true underlying dependencies, we adapt a synthetic stack language dataset from Strobelt et al. (2017) where we know the true dependencies. The vocabulary consists of $\{0, 1, 2, 3, 4, (,)\}$, and the language must match parentheses. Numbers are emitted randomly, but must match the nesting level (the number of open left parentheses). Nesting is limited to depth 4. We follow this grammar and created a training set of 50k sequences, validation/test sets of 5k/5k sequences, with sequence length being 30. We train models to do language modeling on this dataset, where the true dependency for a given target word is the span between the last number and the previous word.

On this dataset, we train SINGLE STREAM DISCRETE TRANSFORMER with proper sparsity regularization (coefficient 0.1). At test time, we use argmax to get discrete attentions, and aggregate attentions to get the receptive field of each prediction. We were able to get precision of 0.959 and recall of 0.920 compared to the ground truth dependencies. In Figure 2, we show an example of the learned receptive field versus the ground truth dependency. On the other hand, if we aggregate attentions of a soft TRANSFORMER model via Eq. 1, the result is much messier, even with the attention sparsity regularizer.

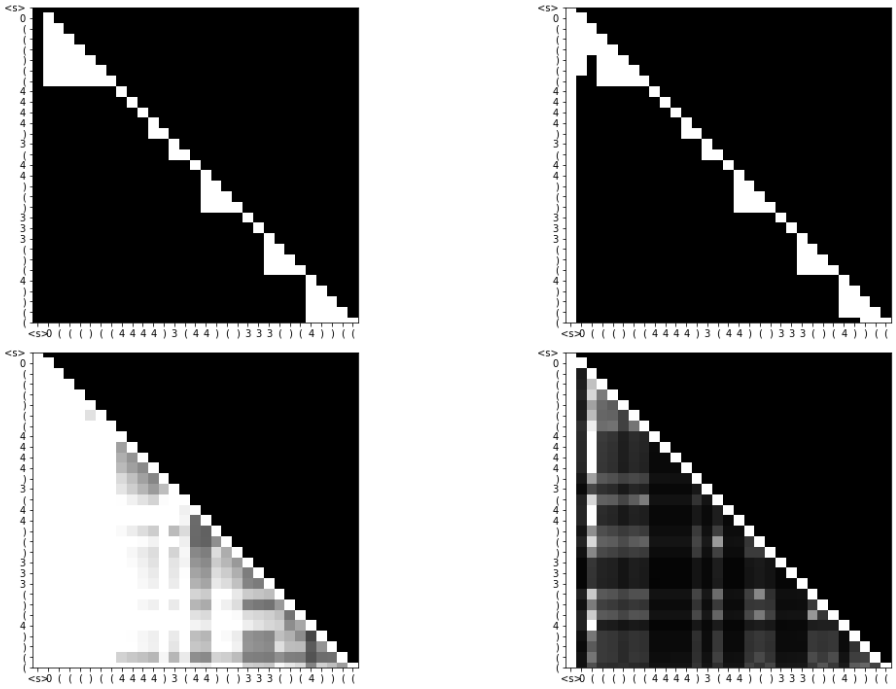


Figure 2: Receptive fields for synthetic data. (Top-L) Ground truth bracketing; (Top-R) Continuous receptive field of sparsity regularized discrete model $r(i, L)$ for rows i; (Bot-L) Attention weighted receptive field of soft transformer; (Bot-R) Attention weighted sparsity regularized receptive field of soft transformer.

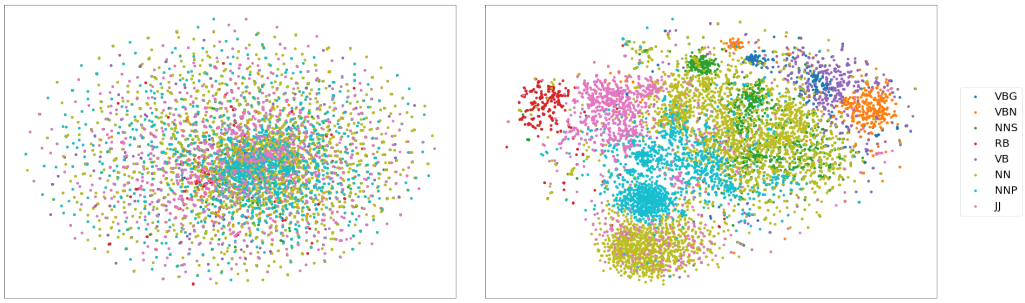
Model	WMT	IWSLT
Vaswani et al. (2017) (base model)	27.3	-
TRANSFORMER	27.3	28.7
SINGLE STREAM DISCRETE TRANSFORMER	26.6	28.5
DISCRETE TRANSFORMER	27.1	28.0

Table 1: BLEU score on WMT English-German (En-De) and IWSLT14. We use WMT16 training data and news2014 as our test set for WMT results.

7.2 MACHINE TRANSLATION

Table 1 shows the BLEU scores of the baselines and our model. Our baseline soft transformer performs identically to the published results. The single-stream discrete transformer SINGLE-STREAM DISCRETE TRANSFORMER performs slightly worse than the soft counterpart TRANSFORMER by 0.7 BLEU points on WMT. On the other hand, our two-stream model DISCRETE TRANSFORMER achieves similar performance as soft baseline, showing that the soft syntax stream helps alleviate model expressivity issues. The results are reversed on the much small IWSLT dataset, with the single-stream model performing better and nearing the results of the soft model.

A benefit of the two-stream model is the separation of the syntactic routing control from the semantic computation. We perform analysis to see how these two components of the model differ. First, we look at a qualitative experiment and visualize the embeddings of the 10,000 most frequent words in the vocabulary using t-SNE (the two streams use independent embeddings). Figure 3 shows the projections for both aspects of the model. We can observe immediately that the embeddings from the syntatic controller network cluster directly by part-of-speech (POS) tags while those from value network do not seem to have a clear pattern. The qualitative nearest neighbors example in Table 3 further confirms that the syntatic controller embeddings cluster by POS tags whereas those from the value network cluster by semantics.



(a) (b)

Figure 3: t-SNE plot of (a) semantic embeddings and (b) syntactic controller embeddings of DISCRETE TRANSFORMER trained on WMT. Datapoints are colored by POS tag assigned by a unigram tagger trained on the WMT train corpus. Best viewed in color.

Model	accuracy	precision	recall	F_1
Baseline	-	72.58	82.14	77.07
TRANSFORMER	89.83	82.25	87.58	84.83
SINGLE STREAM DISCRETE TRANSFORMER	90.70	83.78	88.45	86.05
DISCRETE TRANSFORMER syntactic stream	91.80	85.39	89.51	87.40
DISCRETE TRANSFORMER semantic stream	83.17	73.07	81.28	76.96

Table 2: CoNLL-2000 Chunking. All models are trained with a linear projection over the final encoder layer of fixed WMT model.

To quantify the difference in learned representations we consider utilizing the learned representations. We experiment with using different source encoders from the WMT model as pretrained representations for performing a syntactic chunking task from the CoNLL-2000 dataset. Syntatic chunking consists of dividing a text in syntactically correlated parts of words. For this task we use the pretrained encoder to obtain some vector representation of the source sentence which is then passed through a linear layer to project it to the space of chunk types. The encoder is frozen and only the linear projection layer is trained. For words broken into multiple tokens using the BPE tokenization, we use the vectors from first token.

Table 2 shows the results from different models. The baseline result was obtained by selecting the chunk tag which was most frequently associated with the current part-of-speech tag. We see that both models outperform a standard transformer, and that the syntactic stream of the discrete transformer does the best. Interestingly the semantic stream, which does not have to determine word relations, performs worse then even the baseline model.

Model	Nearest Neighbors
TRANSFORMER	somewhat, slight, little, easily, differently, modest, bit, easy, briefly, rather
SINGLE STREAM DISCRETE TRANSFORMER	somewhat, slight, briefly, little, bit, easily, minor, minimal, barely, partly
DISCRETE TRANSFORMER syntactic stream	somewhat, twice, little, substantially, constantly, almost, considerably, partly, easily, largely
DISCRETE TRANSFORMER semantic stream	somewhat, slight, minor, mild, light, little, easily, growth, significantly, Light

Table 3: Nearest neighbors of word "slightly" using internal representations from different modules. Colors mark POS tags.

8 CONCLUSION

This work presents the discrete transformer, a modification to the transformer to make discrete attention decisions and to separate out dependencies from semantic state value. Experiments show that despite the more structured decisions the model is able to maintain similar performance on standard machine translation benchmarks. Analysis shows that the model separates out syntactic properties and even learns precise decisions on clean data. This style of model opens up the potential for many possible experiments in NLP. Because the model makes hard intermediary decisions the semantic model can be shown to only depend on a subset of the data. This property could be used to check for or remove bias from a model, for instance to ensure that production gendered pronoun does not depend on spurious context. Similarly because the dependencies are predicted separately additional priors or regularization could be used to enforce specific syntactic structure. Finally, this method could be used to train pretrained models that allow for discrete intermediary structure.

REFERENCES

David Alvarez-Melis and Tommi S Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. *arXiv preprint arXiv:1707.01943*, 2017.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer. Proceedings of the second conference on machine translation. In *Proceedings of the Second Conference on Machine Translation*. Association for Computational Linguistics, 2017. URL `http://aclweb.org/anthology/W17-4700`.

Toon Calders, Asim Karim, Faisal Kamiran, Wasif Ali, and Xiangliang Zhang. Controlling attribute effect in linear regression. In *2013 IEEE 13th International Conference on Data Mining*, pp. 71–80. IEEE, 2013.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1721–1730. ACM, 2015.

Mauro Cettolo, Jan Niehues, Sebastian Stuker, Luisa Bentivogli, and Marcello Federico. Report on the 11th IWSLT evaluation campaign. In *Proceedings of IWSLT*, 2014.

William Chan, Navdeep Jaitly, Quoc V Le, and Oriol Vinyals. Listen, attend and spell. *arXiv preprint arXiv:1508.01211*, 2015.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pp. 3504–3512, 2016.

Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. Image-to-markup generation with coarse-to-fine attention. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 980–989. JMLR. org, 2017.

Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander Rush. Latent alignment and variational attention. In *Advances in Neural Information Processing Systems*, pp. 9712–9724, 2018.

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

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. *CoRR*, abs/1902.10186, 2019. URL `http://arxiv.org/abs/1902.10186`.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

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

Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

Gilles Louppe. Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*, 2014.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pp. 1614–1623, 2016.

Vlad Niculae and Mathieu Blondel. A regularized framework for sparse and structured neural attention. In *Advances in Neural Information Processing Systems*, pp. 3338–3348, 2017.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

Ben Peters, Vlad Niculae, and André FT Martins. Interpretable structure induction via sparse attention. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 365–367, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, 1(1):18, 2018.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144. ACM, 2016.

Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

Jake Russin, Jason Jo, and Randall C O'Reilly. Compositional generalization in a deep seq2seq model by separating syntax and semantics. *arXiv preprint arXiv:1904.09708*, 2019.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626, 2017.

Shiv Shankar, Siddhant Garg, and Sunita Sarawagi. Surprisingly easy hard-attention for sequence to sequence learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 640–645, 2018.

Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE transactions on visualization and computer graphics*, 24(1):667–676, 2017.

Inigo Jauregi Unanue, Ehsan Zare Borzeshi, and Massimo Piccardi. A shared attention mechanism for interpretation of neural automatic post-editing systems. *arXiv preprint arXiv:1807.00248*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *Proceedings of NIPS*, 2017.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057, 2015.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.