Symbolic Autoencoding for Self-Supervised Sequence Learning

Mohammad Hossein Amani [1] **Nicolas Mario Baldwin*** [1] **Amin Mansouri*** [1 2] **Martin Josifoski** [1]
Maxime Peyrard [3] **Robert West** [1]

Abstract

Traditional language models(LMs) excel at next-token prediction in text sequences but often struggle with transduction tasks involving distinct symbolic systems, particularly when parallel data is scarce or nonexistent. This issue is even more pronounced in domains dealing with complex, non-natural language sequences, such as audio signals, protein structures, or biological sequences, where the strengths of LMs in natural language do not directly translate. To address this challenge, we introduce *symbolic autoencoding* (ΣAE), a self-supervised framework designed to exploit the wealth of non-parallel data alongside limited parallel data. ΣAE integrates two generative models via a discrete bottleneck layer, optimizing the entire system end-to-end by minimizing unsupervised reconstruction loss for all data such that the sequence generated at the discrete bottleneck can be read out as the transduced input sequence, and separately optimizing the two models with supervised loss on the subset of labeled parallel data. To allow optimization of the models in the presence of discrete symbols, we use a family of straight-through gradient estimators. We demonstrate the effectiveness of ΣAE on four sequence-to-sequence transduction tasks, showing that it significantly outperforms strong baselines in weakly supervised settings.

1. Introduction and Preliminaries

The field of artificial intelligence has undergone a remarkable transformation in recent years, propelled by the rise of powerful language models. At the heart of this success are sequence-to-sequence (seq2seq) transducers, a class of models trained to infer the mapping M between two symbolic systems X and Z such that $Z = M(X)$. Recent large language models display striking emergent abilities to perform many such mappings after exposure to massive and diverse textual data. However, they fail when one or both language systems are scarce or nonexistent in the training data, or when the mapping function deviates significantly from patterns present during training (Magueresse et al., 2020; Lample & Conneau, 2019; Joshi et al., 2020).

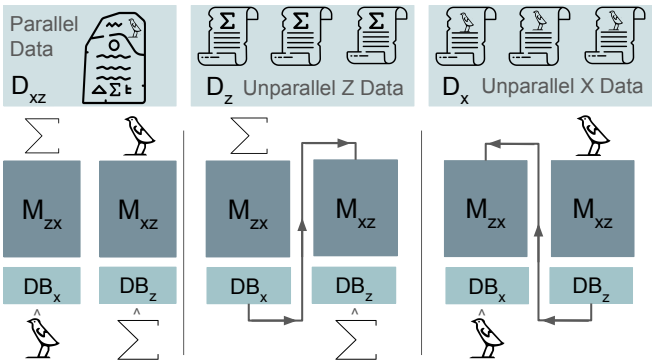

Figure 1. Illustration of the abstract flow of data in the symbolic autoencoding (ΣAE) framework, exemplified with the Rosetta Stone problem. Two sequence-to-sequence models (M_{xz} and M_{zx}) are trained with both parallel data (the Rosetta Stone) through next-token prediction and unparallel data through connecting the models with a discrete bottleneck layer (DB_x and DB_z) to autoencode each language using the other as its hidden representation.

An epitome of such language systems is the ancient Egyptian hieroglyphs, a system of writing used in ancient Egypt. Although Egyptian hieroglyphs were abundant in ancient Egyptian papyri, they remained a mystery until the discovery of the Rosetta Stone in 1799, which provided the key to unlocking the secrets of the hieroglyphs (Budge, 1913). The stone slab bears the same text inscribed in three distinct scripts: Egyptian hieroglyphs, Demotic script, and ancient Greek. This limited parallel dataset, a mere 27 lines of text, together with abundant unparallel text in the three scripts was sufficient to guide researchers in understanding the full translation M between ancient Egyptian hieroglyphs (X) and ancient Greek (Z). The elegant solution to this historic challenge leads us to pose the question: **how can we train models to automatically leverage the information in the unparallel data in symbolic systems X and Z to help with learning the mapping M between them?**

*Equal contribution [1]Department of Computer Science, EPFL, Lausanne, Switzerland [2]Mila, Quebec AI Institute and University of Montreal, Montreal, Canada [3]Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG. Correspondence to: Mohammad Hossein Amani <mh.amani1998@gmail.com>.

In this work, we address this question by introducing *symbolic autoencoding* (ΣAE), a novel method that simultaneously learns the mapping M_{xz} from X to Z and the mapping M_{zx} from Z to X, symmetrically, using both parallel and unparallel data. This is achieved by orchestrating several losses that reuse M_{xz} and M_{zx} with varying inputs and outputs. First, two supervised losses are used to tune M_{xz} and M_{zx} on the scarce parallel data. Given the scarcity of parallel data, these alone are insufficient to learn the mappings. Therefore, we introduce two additional autoencoding losses. One autoencoder reconstructs $z \in Z$ after encoding it discretely into an unknown $x \in X$ via the path $Z \xrightarrow{M_{zx}} X \xrightarrow{M_{xz}} Z$, where M_{zx} is the encoder and M_{xz} is the decoder. The connection between the two models is implemented by a *discrete bottleneck* (DB), a differentiable mechanism that binds two sequence-to-sequence models into an end-to-end, fully differentiable sequence-to-sequence-to-sequence model. The discrete bottleneck serves as the differentiable glue between M_{xz} and M_{zx} by allowing gradients to pass through the discrete latent space using straight-through estimator (Bengio et al., 2013) and a general family of sampling/quantization methods. Symmetrically, the second autoencoder reconstructs x after encoding it into z. These two autoencoding losses utilize non-parallel data, treating X as the hidden representation for the autoencoding of Z, and vice versa. The supervised losses further ensure that the hidden symbolic systems are grounded in the parallel data and not arbitrary languages.Fig. 1 illustrates the abstract flow of data in the ΣAE framework.

We evaluate ΣAE on four sequence-to-sequence transduction tasks, demonstrating that it significantly outperforms the three conventional strategies in weakly supervised settings: fine-tuning a pre-trained language model using the limited parallel data available, using in-context learning to adapt the model to the new task, or the more direct approach which is training a model from scratch only on supervised data. Our main contributions are the following:

- We introduce ΣAE as a framework for connecting two seq2seq models via DB and training them using gradient-based optimization.
- We unify celebrated methods such as VQ-VAE and the Gumbel-Softmax trick as different design choices for the quantization function in DB and compare their performances in our experiments.
- We empirically benchmark the performance of ΣAE over traditional supervised baselines using both synthetic and real-world data.

To facilitate further research, we open-source code and data.

Related work. Our work intersects with several key areas in unsupervised and weakly supervised learning through discrete representations.

Baziotis et al. (2019) connected two encoder-decoder models via a hidden sequence layer, employing a reconstruction loss and a language model prior loss for unsupervised text compression. Kaiser & Bengio (2018) explored semantic hashing (Salakhutdinov & Hinton, 2009) and the Gumbel-Softmax trick (Jang et al., 2017) for generating interpretable, discrete encodings. Similarly, Fortuin et al. (2019) investigated training with discrete bottlenecks and examined the use of continuous paths alongside discrete ones.
Zhu et al. (2017) and He et al. (2016) enforced consistency across translation tasks, with Zhu et al. (2017) using adversarial networks and He et al. (2016) employing reinforcement learning to update the models. Our work introduces a third approach, leveraging straight-through gradient estimators to train models end-to-end. Furthermore, numerous studies have focused on the discretization of elements and representations in neural networks (Liu et al., 2022; 2021; Tamkin et al., 2023; Peng et al., 2018; Maddison et al., 2016). Our proposed solution also parallels the technique of back-translation (Sennrich et al., 2015; Çaglar Gülçehre et al., 2015; 2017), which typically involves training an intermediate system on parallel data to translate target monolingual data into the source language, thereby generating synthetic parallel corpora for further training (Edunov et al., 2018). The ΣAE framework is akin to an online version of back-translation, where the intermediate system is continuously improving without storing synthetic sequences.

2. ΣAE Framework

2.1. Discrete Bottleneck

In our setup a DB provides two essential outputs:

- **Probability vector** $\mathbf{s}$ represents a discrete distribution over tokens, facilitating training with negative log-likelihood loss when labels are available. (supervised training)
- **Quantized vector** $\mathbf{v}_q$ serves as input for subsequent models or layers, such as the decoder in reconstruction tasks when labels are not available. (unsupervised training)

Thus, DB can be described as a function $\mathbf{s}, \mathbf{v}_q = \mathrm{DB}(\mathbf{v})$, where $\mathbf{s} \in [0,1]^{\|V\|}$ and $\sum_{i=1}^{\|V\|} \mathbf{s}_i = 1$ with $|V|$ as the size of the vocabulary. The discrete nature of the DB implies that the quantized vector belongs to a finite discrete domain $\mathbf{v}_q \in D$, like a dictionary of embeddings $D = \{D[i]\}_{i=1}^{|V|}$. This discrete computation within the DB introduces a point of non-differentiability, necessitating the use of surrogate gradients to enable gradient-based optimization.

2.1.1. Discrete bottleneck implementations

The DB allows us to see the celebrated methods such as VQ-VAE and the Gumbel-Softmax reparameterization trick

as different implementations of the same concept. We can classify DBs into two categories: **probability-based** and **embedding-based**.

Embedding based DB. Here the probability vector $\mathbf{s}$ is a function of the dictionary embeddings, $\mathbf{s} = S(\cdot; D)$. For instance, this scoring function S can be the softmax function of any distance metric between the input vector $\mathbf{v}$ and the dictionary embeddings $D[i]$, with the quantized vector $\mathbf{v}_q$ being the closest dictionary embedding to the input vector $\mathbf{v}$ according to that metric:

$$\mathbf{l}[i] = \|\mathbf{v} - D[i]\|, \ \mathbf{v}_q = D\left[\arg\min_i \mathbf{l}[i]\right], \ \mathbf{s}[i] = S(-\mathbf{l}[i])$$

with $\mathbf{l}$ representing the vector of distances between $\mathbf{v}$ and dictionary vectors $D[i]$. In backpropogation, gradients are directly passed from $\mathbf{v}_q$ to $\mathbf{v}$ using the assignment $\mathbf{v}_q \leftarrow \mathbf{v}_q + \mathbf{v} - \text{sg}(\mathbf{v})$ in the computation graph, where sg denotes the stop-gradient operation (van den Oord et al., 2017; Bengio et al., 2013). In our experiments we focus on the **vector-quantized DB (VQ DB)** similar to VQ-VAE, with the distance metric being the Euclidean distance.

Probability-based DB. Here the score function does not depend on the dictionary embeddings $\mathbf{s} = S(x)$, and the quantized vector $\mathbf{v}_q$ is computed by decoding/sampling from the score vector $\mathbf{v}_q = D[\text{decode}(\mathbf{s})]$. Therefore, to concretely implement a probability-based DB, we need to define a score function $S(\cdot)$ and a sampling method. In this work we take S to be a softmax function, and use maximum likelihood decoding and categorical sampling for decoding:

- **Softmax DB** uses maximum likelihood decoding, i.e., the quantized vector $\mathbf{v}_q$ corresponds to the most likely token in the dictionary:

$$\mathbf{s}[i] = \frac{\exp(\mathbf{v}[i])}{\sum_{j=1}^{|V|}\exp(\mathbf{v}[j])}, \quad \mathbf{v}_q = D\left[\arg\max_i \mathbf{s}[i]\right]$$

- **Gumbel DB** uses categorical sampling for decoding:

$$\mathbf{s}[i] = \frac{\exp(\mathbf{v}[i] + g_i)}{\sum_{j=1}^{|V|}\exp(\mathbf{v}[j] + g_j)}, \quad \mathbf{v}_q = D\left[\arg\max_i \mathbf{s}[i]\right]$$

  Here g_i is a sample from the Gumbel distribution, i.e. $g_i = -\log(-\log(u_i))$ where $u_i \sim \text{Uniform}(0, 1)$, using the Gumbel reparameterization trick to translate the sampling to taking the argmax of noisy probabilities (Jang et al., 2017).

Crucially, during the backward pass, gradients are passed to $\mathbf{s}$ as if $\mathbf{v}_q$ was the soft average of dictionary embeddings. This is expressed by assigning $\mathbf{v}_q \leftarrow \mathbf{v}_q + \sum_{i=1}^{|V|}\mathbf{s}[i]D[i] - \text{sg}(\sum_{i=1}^{|V|}\mathbf{s}[i]D[i])$ in an automatic differentiation library.

2.2. Training Models with DB Head

We incorporate a DB layer into each seq2seq model: DB_x to M_{zx} and DB_z to M_{xz}, enabling both separate and joint training modes.

For parallel training data $(x, z) \in D_{xz}$ we do a **supervised training** step similar to common seq2seq training. Given input sequence x and target sequence until step t, $z^{<t}$, the model M_{xz} predicts a probability vector $\mathbf{s}_z^t$ for the t-th token z^t and recieves a loss (similarly for predicting the x sequence):

$$\mathbf{s}_z^t, \mathbf{v}_z^t = \text{DB}_z(M_{xz}(x, z^{<t})), \ \mathcal{L}_{xz} = -\sum_t \log \mathbf{s}_z^t[z^t]$$

Given unlabeled data ($x \in D_x$ or $z \in D_z$), the models generate a latent sequence of quantized vectors ($\mathbf{v}_x^{<T_x} = \{\mathbf{v}_x^t\}_{t=0}^{T_x}$):

$$\mathbf{s}_x^t, \mathbf{v}_x^t = \text{DB}_x(M_{zx}(z, \mathbf{v}_x^{<t}))$$

These vectors are then used to reconstruct the original input:

$$\mathbf{s}_z^t, \mathbf{v}_z^t = \text{DB}_z(M_{xz}(\mathbf{v}_x^{<T_x}, z^{<t}))$$

using as the reconstruction loss $\mathcal{L}_{zxz} = -\sum_t \log \mathbf{s}_z^t[z^t]$. Similar steps are followed for the x sequence. We call these **X Reconstruction** and **Z Reconstruction** modes, where we use unparallel data, D_x or D_z, to minimize reconstruction losses $\mathcal{L}_{xzx}$ or $\mathcal{L}_{zxz}$.

To navigate this multi-objective optimization problem, we propose three scheduling strategies: **Joint Training** involves randomly selecting a batch from D_{xz}, D_x, or D_z at each iteration and training in the corresponding mode. **Unsupervised Pretraining with Supervised Finetuning** starts with training on D_x and D_z until convergence, followed by fine-tuning on D_{xz}. Conversely, **Supervised Pretraining with Unsupervised Finetuning** trains on D_{xz} until convergence, then shifts to fine-tuning on D_x and D_z.

2.3. Hidden Sequence Collapse in Seq2Seq Models

In symbolic autoencoding, the encoder seq2seq model autoregressively generates hidden tokens until an End-of-Sequence (EOS) token or a maximum length is reached. This process involves a discrete decision about when to halt generation, for which the model never receives gradient feedback. Specifically, in unsupervised training, the loss gradient doesn't directly inform the model that mistakenly assigning a high likelihood to EOS has a penalty beyond the negative log likelihood loss: it can prematurely stop the entire sequence generation. In our early autoencoder trainings we empirically observed that the models tended to rely excessively on the first token of the hidden representation, leading to underutilization of subsequent tokens. This led us to develop the following solution.

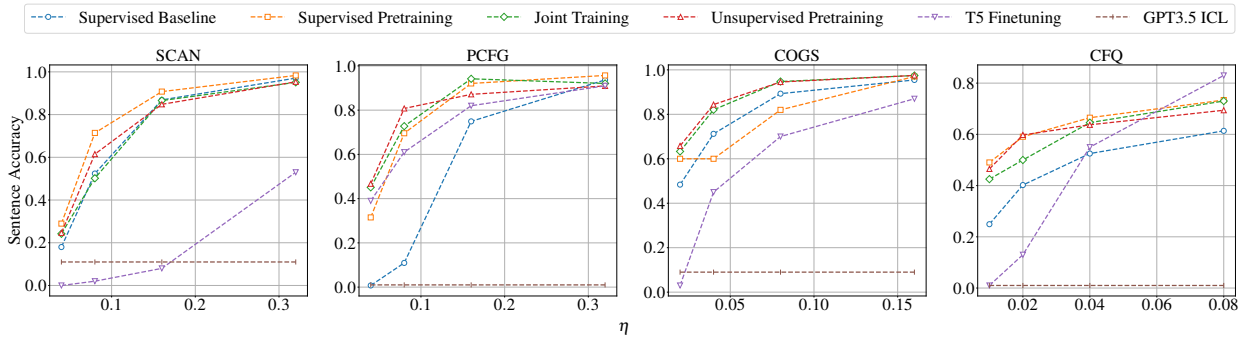

Figure 2. Results for Softmax Discrete Bottleneck – Z Autoregressive Sentence Accuracy per Supervision Ratio (η). At least one training method with the ΣAE framework consistently outperforms the pretrained and in-context learning baselines except on the CFQ dataset at 8% supervision ratio.

EOS Soft-Masking – Gradient Approximation for Halting the Generation. In unsupervised training mode, hidden sequences in a batch can have varying halting points as the generation typically continues until the maximum length is reached or all samples have produce an EOS token. Tokens generated after the EOS are masked out using a binary mask $\mathbf{m}$ of size T (the number of tokens), where each $\mathbf{m}[i]$ is 1 if the EOS token has not been generated and 0 otherwise. Applying the mask to the quantized vectors $\mathbf{v}_q^{<T}$ during the forward pass enforces a halting mechanism by setting vectors post-EOS to a padding embedding, $\mathbf{v}_q \leftarrow \mathbf{v}_q \odot \mathbf{m} + D[\text{<PAD>}] \odot (1 - \mathbf{m})$, thereby terminating the sequence generation. The challenge arises during the backward pass, as this mask is a non-differentiable output of the forward computation. To address this, we propose a gradient approximation for $\mathbf{m}$ that allows the model to learn the EOS effect through a feedback mechanism. To mitigate autoregressive collapse, we pass the gradients through $\mathbf{m}$ to $\mathbb{P}(O_k = \text{<EOS>})$ as if $\mathbb{E}[\mathbf{m}[i]] = \prod_{k=1}^{i-1}(1 - \mathbb{P}(O_k = \text{<EOS>}))$ had been the masking matrix in the forward computation. This approximation provides direct feedback on the EOS effect by simply assigning $\mathbf{m} \leftarrow \mathbf{m} + \mathbb{E}[\mathbf{m}] - \text{sg}(\mathbb{E}[\mathbf{m}])$. The derivation of this approximation is detailed in Appendix A.2.

3. Experiments

Datasets. For our experiments, we utilized four seq2seq datasets: **SCAN** (Lake & Baroni, 2017), **PCFG SET** (Hupkes et al., 2019), **CFQ** (Keysers et al., 2019), and **COGS** (Kim & Linzen, 2020), chosen for their compositional complexity, controlled environments, and precise accuracy measures. We evaluated the framework on the aforementioned datasets, focusing on sentence accuracy (SA) and token accuracy (TA). Additional performance metrics are discussed in the appendix in Section A.5. More details on the datasets are provided in Section A.3.

Baselines. In our experiments, we compare the performance of the ΣAE framework against the following baselines: (1) **Supervised Fine-tuning of a Pretrained Model (T5 large)**, where a pretrained T5 model is fine-tuned on the available parallel data; (2) **In-context Learning (ICL) with a Large Language Model (GPT-3.5)**, which utilizes GPT-3.5 to perform tasks based on given context without explicit fine-tuning; and (3) **Supervised Training from Scratch**, where a model is trained from scratch on the available parallel data. Further details on the tasks, model architecture, and hyperparameters are provided in Section A.4

Experimental Results. To show the feasibility of symbolic autoencoding with straight-through gradients updates we performed an unsupervised autoencoding reconstruction experiment for each dataset and DB and observed that the models successfully learned a compression of the input sequences, as shown in Table 2. The results are further detailed in Sec. A.7.1.

In the weakly supervised task, we simulated a Rosetta Stone-like scenario with a mix of parallel and unparallel data, varying the ratio of parallel data (η) to assess the framework's ability to balance and integrate supervised and unsupervised losses. Notably, the unsupervised task is a special case of the weakly supervised task where $\eta = 0$. Results for the Softmax DB are detailed in Figure 2.

Our experiments demonstrated that the ΣAE framework can efficiently utilize small amounts of parallel data to improve performance on larger unparallel datasets. At each supervision ratio η, one of our scheduling methods from Section 2.1 consistently outperformed the supervised baselines. As expected, model accuracy improved with increased supervised data, narrowing the performance gap as accuracies converged to their maxima. An exception was observed in the CFQ dataset at an 8% supervision ratio, where fine-tuning the T5-large model outperformed our methods. This is likely due to the CFQ dataset's closer resemblance to natural language question answering tasks, benefiting the T5 model, which is pretrained on similar tasks. Additional remarks on training dynamics and learning behavior are provided in Section A.6. A detailed analysis of the results and the full set of performance metrics, including other DBs, are presented in Section A.7.2.

References

Bastings, J., Baroni, M., Weston, J., Cho, K., and Kiela, D. Jump to better conclusions: Scan both left and right. *ArXiv*, abs/1809.04640, 2018.

Baziotis, C., Androutsopoulos, I., Konstas, I., and Potamianos, A. Seqˆ3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *North American Chapter of the Association for Computational Linguistics*, 2019.

Bengio, Y., Léonard, N., and Courville, A. C. Estimating or propagating gradients through stochastic neurons for conditional computation. *ArXiv*, abs/1308.3432, 2013.

Budge, S. E. A. W. *The Rosetta stone*. London: British Museum, 1913.

Edunov, S., Ott, M., Auli, M., and Grangier, D. Understanding back-translation at scale, 2018.

Fortuin, V., Hüser, M., Locatello, F., Strathmann, H., and Rätsch, G. Som-vae: Interpretable discrete representation learning on time series. In *International Conference on Learning Representations*, 2019.

He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T.-Y., and Ma, W.-Y. Dual learning for machine translation. In *Neural Information Processing Systems*, 2016.

Hupkes, D., Dankers, V., Mul, M., and Bruni, E. Compositionality decomposed: How do neural networks generalise? *J. Artif. Intell. Res.*, 67:757–795, 2019.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., and Choudhury, M. The state and fate of linguistic diversity and inclusion in the NLP world. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6282–6293, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.560.

Kaiser, L. and Bengio, S. Discrete autoencoders for sequence models. *ArXiv*, abs/1801.09797, 2018.

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., and Bousquet, O. Measuring compositional generalization: A comprehensive method on realistic data. *ArXiv*, abs/1912.09713, 2019.

Kim, N. and Linzen, T. Cogs: A compositional generalization challenge based on semantic interpretation. *ArXiv*, abs/2010.05465, 2020.

Lake, B. M. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, 2017.

Lample, G. and Conneau, A. Cross-lingual language model pretraining. *ArXiv*, abs/1901.07291, 2019.

Liu, D., Lamb, A., Kawaguchi, K., Goyal, A., Sun, C., Mozer, M. C., and Bengio, Y. Discrete-valued neural communication. In *Neural Information Processing Systems*, 2021.

Liu, D., Lamb, A., Ji, X., Notsawo, P. J. T., Mozer, M. C., Bengio, Y., and Kawaguchi, K. Adaptive discrete communication bottlenecks with dynamic vector quantization. *ArXiv*, abs/2202.01334, 2022.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. *ArXiv*, abs/1611.00712, 2016.

Magueresse, A., Carles, V., and Heetderks, E. Low-resource languages: A review of past work and future challenges. *ArXiv*, abs/2006.07264, 2020.

Peng, H., Thomson, S., and Smith, N. A. Backpropagating through structured argmax using a spigot. *ArXiv*, abs/1805.04658, 2018.

Salakhutdinov, R. and Hinton, G. E. Semantic hashing. *Int. J. Approx. Reason.*, 50:969–978, 2009.

Sennrich, R., Haddow, B., and Birch, A. Improving neural machine translation models with monolingual data. *ArXiv*, abs/1511.06709, 2015.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958, 2014.

Tamkin, A., Taufeeque, M., and Goodman, N. D. Codebook features: Sparse and discrete interpretability for neural networks. *ArXiv*, abs/2310.17230, 2023.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6306–6315, 2017.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251, 2017.

Çaglar Gülçehre, Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H.-C., Bougares, F., Schwenk, H., and Bengio, Y. On using monolingual corpora in neural machine translation. *ArXiv*, abs/1503.03535, 2015.

Çaglar Gülçehre, Firat, O., Xu, K., Cho, K., and Bengio, Y. On integrating a language model into neural machine translation. *Comput. Speech Lang.*, 45:137–148, 2017.

A. Appendix

A.1. Remarks on ΣAE framework

For all our straight-through gradient estimations, as training progresses, models become more confident in their predictions, resulting in more polarized score distributions. This polarization helps the models identify the most likely token with increasing certainty, making the scores sparser and improving the accuracy of gradient approximations.

While we only use symbolic autoencoding in reconstruction setups, the framework is adaptable to additional models and data sources. For instance, one could imagine models M_{zy}, M_{yx}, etc., each with their own supervised and reconstruction losses (e.g., $\mathcal{L}_{zy}, \mathcal{L}_{yxz}, \mathcal{L}_{xyx}$, etc.) to be optimized. Unlike some multi-task scenarios where individual tasks may appear independent or unrelated, in the ΣAE framework, improvement in one task can directly benefit others, creating a synergy that enhances overall performance.

A.2. EOS Gradient Approximation

The EOS collapse phenomena can be explained by the model's lack of understanding of the EOS token's impact. Without explicit feedback, the model does not learn the importance of distributing information across the entire sequence. Instead, it packs all information into the first token to ensure it reaches the decoder robustly. This behavior is akin to how models learn more robust representations under dropout conditions (Srivastava et al., 2014), where information is concentrated into fewer units.

In the ΣAE framework, we inform the model of the halting effect of the EOS token by approximating a gradient for the mask $\mathbf{m}$, which masks the tokens appearing after the first EOS token. This approximation is crucial for the model to learn the halting effect of the EOS token, essential for generating accurate sequences.

The $\mathbf{m}$ is 1 if the EOS token has not been generated and 0 otherwise:

$$\mathbf{m}[i] = \begin{cases} 1 & \text{if } \mathbf{m}[i-1] = 1 \text{ and } O_{i-1} \neq \text{<EOS>} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Where O represents the output sequence generated by the model, whether in X or Z.

Hence, the binary random vector $\mathbf{m}$ is defined as:

$$P(\mathbf{m}[i] = 1) = \left(1 - \mathbb{P}(O_{(i-1)} = \text{<EOS>})\right) \mathbb{P}(\mathbf{m}[i-1] = 1) = \prod_{k=1}^{i-1} \left(1 - \mathbb{P}(O_k = \text{<EOS>})\right). \tag{2}$$

Therefore the expected value of $\mathbf{m}$ is:

$$\mathbb{E}[\mathbf{m}[i]] = \prod_{k=1}^{i-1} \left(1 - \mathbb{P}(O_k = \text{<EOS>})\right) \tag{3}$$

Our ablation studies showed that unsupervised training often failed due to hidden state collapse when this approximation was not used. Without this gradient approximation, the model struggled to learn effectively, highlighting the importance of this technique for successful training.

As training progresses, models become more confident in correctly predicting the EOS token, leading to more polarized probabilities. This makes the expected mask $\mathbb{E}[\mathbf{m}]$ a better approximate the true mask, thereby improving the accuracy of our approximation.

A.3. Dataset Description and Examples

We evaluated the ΣAE framework on four diverse datasets: SCAN, PCFG SET, CFQ, and COGS.

- SCAN (Lake & Baroni, 2017) is a simple language-driven navigation instruction task designed to evaluate the ability of neural models to learn compositional commands.

Table 1. Example of Samples from Different Datasets

Dataset	Sample	Train set size	Parallel portion
SCAN	**X:** look right thrice after run left **Z:** I_TURN_LEFT I_RUN I_TURN_RIGHT I_LOOK I_TURN_RIGHT I_LOOK I_TURN_RIGHT I_LOOK	13382	1% to 8%
PCFG SET	**X:** echo append append E18 C13 , L18 M17 , R1 L1 Y1 T18 J18 **Z:** E18 C13 L18 M17 R1 L1 Y1 T18 J18 J18	65734	4% to 32%
CFQ	**X:** Who influenced M1 's cinematographer , writer , and editor **Z:** SELECT DISTINCT ?x0 WHERE ?x0 a ns:people.person. ?x0 ns:influence.influence_node.influenced ?x1. ?x1 ns:film.cinematographer.film M1. ?x1 ns:film.editor.film M1. ?x1 ns:film.writer.film M1.	76594	2% to 16%
COGS	**X:** Olivia rolled Liam. **Z:** roll . agent (x_1 , Olivia) AND roll . theme (x_1 , Liam)	24155	1% to 8%

- PCFG SET (Hupkes et al., 2019) is a synthetic dataset generated using probabilistic context-free grammars, aimed at testing the systematic generalization of models.
- CFQ (Keysers et al., 2019) is a large-scale dataset of complex natural language questions and their corresponding SPARQL query against the Freebase knowledge base designed to measure the compositional generalization capabilities of semantic parsing models, with questions constructed to reflect the compositional structure of Freebase.
- COGS (Kim & Linzen, 2020): COGS is a dataset for evaluating the generalization of semantic parsing models to novel linguistic structures, emphasizing the model's ability to generalize from given sentences to new sentences that have similar syntactic structures but different lexical items or phrasal constructions.

These datasets were chosen for their controlled environments and precise accuracy measures, making them ideal for evaluating the framework's performance. Examples of samples from each dataset are provided in Table 1.

The selection of these datasets ensures a comprehensive and nuanced evaluation of the ΣAE framework. They facilitate direct evaluation of our approach, avoiding reliance on proxy metrics often used with larger datasets. Here, the mapping from X to Z is unique and non-reversible, with Z typically being the longer sequence, serving as a reliable ground truth for X. Our study diverges from the typical use of these datasets for compositional generalization. Instead of focusing on out-of-distribution testing, we emphasize in-distribution performance assessment. We also conduct a bidirectional evaluation of both M_{xz} and M_{zx} models, reflecting realistic seq2seq model applications where translation in both directions holds equal significance, in line with the suggestions of (Bastings et al., 2018).

A.4. Details on Tasks, Model Architecture, and Hyperparameters

We conducted two sets of experiments on each dataset:

- **Unsupervised Training**: In this scenario, we only have access to unparallel data. The primary goal is to reconstruct Z from a hidden discrete sequence. The framework matches the dictionary size and the maximum sequence length of the hidden representation to those of X. This setup evaluates the ΣAE framework's ability to compress the input sequence into a shorter sequence and accurately reconstruct it.
- **Weakly-supervised Training**: This scenario simulates the Rosetta Stone problem, where a small portion of the data is parallel, and the rest is unparallel. The objective is to leverage both parallel and unparallel data by minimizing unsupervised losses ($\mathcal{L}_{zxz}$ and $\mathcal{L}_{xzx}$) and supervised losses ($\mathcal{L}_{zx}$ and $\mathcal{L}_{xz}$). We conduct experiments for each dataset and DB implementation, varying the supervision ratio $\eta = \frac{|D_{xz}|}{|D_{xz}|+|D_x|+|D_z|}$. This allows us to assess how effectively the framework uses limited parallel data to improve performance on larger unparallel datasets.

In our experiments with the ΣAE framework, we adopted a standardized model architecture and hyperparameter setting across all tasks to maintain consistency and focus on the framework's effectiveness. We utilized a six-layer transformer encoder–decoder model for M_{xz} and M_{zx}, with 8 attention heads and a hidden size of 512. The model was trained using the Adam optimizer with learning rate reduction on loss plateau. We used greedy decoding consistently for all tasks, simplifying the decoding process and ensuring uniformity across experiments.

Model learning rates were manually chosen on the order of 10^{-3} or 10^{-4}, to ensure a decrease in loss during the early stages of training. Hyperparameters were not extensively tuned. For each task, the same hyperparameters were used across different supervision ratios which are available in our configuration files in the code. This uniform approach underscores the framework's robustness, although we acknowledge that more nuanced tuning and regularization might yield higher performance.

In both unsupervised and supervised finetuning after pretraining approaches, a gradual curriculum shift is employed rather than an abrupt change. This involves slowly altering the probability distribution of the 'three-sided coin' used for batch selection in joint training, to transition smoothly from the initial training phase to the subsequent finetuning phase.

A.5. Evaluation Metrics

In assessing the performance of the ΣAE framework, we measured two distinct metrics: **sentence accuracy (SA)** and **token accuracy (TA)**. These metrics are designed to provide both a holistic and a detailed view of the model's capabilities. Sentence accuracy (SA) for a sample is counted as 1 if the entire sentence is correctly generated. Token accuracy (TA) is a more granular measure, where correctness of each predicted token in all sentences are measured separately. This metric allows for partial credit within a sentence, providing a finer understanding of the model's performance at the token level.

The token accuracy can be measured with two methods: We can *teacher-force* the correct previous tokens (as per the ground truth) to the model and measure its accuracy in predicting the next token. Alternatively, the model's previous outputs (which may or may not be correct) can be used as inputs for generating subsequent tokens. This *autoregressive* approach is generally more challenging than teacher-forcing.

Each X has a unique corresponding Z, simplifying the assessment of accuracy in this direction, therefore, evaluating M_{xz} performance is simply done by examining the *Autoregressive Z TA/SA*, directly measuring the model's capability to generate accurate Z sequences. For a given Z, however, there could be multiple valid X sequences. Therefore, to evaluate M_{zx}, we utilize the *Teacher-forced X TA*, which restricts the range of correct X sequences for end tokens. Another approach is the *Reconstruction Z TA/SA*, where a model M_{xz} maps a generated sequence $\hat{x}$ back to Z, and the accuracy of this reconstructed sequence serves as a proxy for the correctness of $\hat{x}$.

A.6. Remarks on Experimental Results

We note that the VQ DB faced a peculiar issue of numerical instability on the SCAN dataset after extended training periods (+500 epochs). This instability was addressed through weight clipping, suggesting that while ΣAE offers substantial benefits, optimizing stability and accuracy across different data representations and tasks may require tailored adjustments. These insights into performance variations across X and Z spaces not only highlight the framework's broad applicability but also pinpoint areas for future refinement to maximize the ΣAE framework's effectiveness.

A.7. Experiment Results

A.7.1. Unsupervised Training Results

In the unsupervised task, we trained the discrete autoencoder to compress and reconstruct Z sequences without any supervised signal, evaluating the learnability of the discrete bottleneck using straight-through gradients. The results, summarized in Table 2, show that the Softmax DB achieved over 98% token accuracy on the SCAN, CFQ, and COGS datasets. Both the Gumbel and VQ DBs demonstrated similar effectiveness, indicating robustness in discrete autoencoding with straight-through gradients for sequence learning tasks. An exception to the high performance was the PCFG SET reconstruction task, where model performances were notably lower. This variation may be attributed to the unique symbolic nature of variables within the PCFG SET task, where basic tokenization assigns distinct representations to symbolically equivalent variables, leading to observed performance discrepancies.

A.7.2. Weakly Supervised Training Results

In the Z space, the Softmax DB consistently surpassed supervised baselines, significantly enhancing token and sentence accuracy across all datasets. For instance, with only 8% supervision on the PCFG SET dataset, token accuracy improved from below 15% to above 80%. While the Gumbel DB generally showed noisier training and slightly weaker performance, it still outperformed supervised baselines in most scenarios, except for a minor shortfall in the COGS dataset at a 16%

Table 2. Table of Test Autoregressive token accuracy (Z) (top) and Sentence Accuracy (Z) (bottom) on the unsupervised autoencoding task (Z reconstruction). A high accuracy is achieved across all datasets, showing the feasibility of learning discrete representations with gradient-based methods.

	SCAN	PCFG	COGS	CFQ
Softmax DB	1.00	0.74	0.98	0.99
	0.96	0.31	0.55	0.69
Gumbel DB	0.98	0.75	0.98	0.99
	0.74	0.36	0.51	0.43
VQ DB	1.00	0.44	0.94	0.90
	0.93	0.00	0.03	0.00

supervision ratio. The VQ DB, despite showing a slight weaker performance in supervised baselines, improved the training similar to the Softmax and Gumbel DBs, achieving over 20% token accuracy on CFQ dataset at 2% supervision ratio.

While no single Discrete Bottleneck or scheduling method universally outperforms others across all datasets and supervision ratios, for every dataset and η value, at least one of our scheduling methods consistently surpasses the baseline performance. In other words, training within the ΣAE paradigm always enhances performance, though the optimal choice of the scheduling strategy depends on the task.

The ΣAE framework's impact extends into the X space, where the Softmax, Gumbel, and VQ DBs exhibit performance boosts. Notably, the exception to this trend occurs with teacher-forced token accuracy in the SCAN dataset for the Softmax DB, indicating a unique challenge in this specific setting.

For all our experiments, we computed 95% confidence intervals via bootstrapped resampling of the test set, however they are too small to be visible on the plots. This performance analysis underscores the ΣAE framework's versatility and its capacity to leverage both unsupervised and weakly supervised data to enhance model training and performance across diverse seq2seq tasks.

We only measure the ICL and supervised finetuning of T5 baselines for Autoregressive Z TA and SA, as the teacher-forced X TA is not applicable to these baselines. The ICL baseline is a flat line with a fixed number of in-context samples (20) and the supervised finetuning of T5 is a single point at 100% supervision ratio.

We present the results of our experiments in the following tables. For Softmax discrete bottleneck, we present the results in the following tables:

- Table 3 Shows the performance of the Softmax DB on Autoregressive Z token accuracy, from test inputs
- Table 4 Shows the performance of the Softmax DB on Autoregressive Z sentence accuracy, from test inputs
- Table 5 shows the performance of the Softmax DB on the Autoregressive Z reconstruction token accuracy, after mapping to a hidden X
- Table 6 shows the performance of the Softmax DB on the Autoregressive Z reconstruction sentence accuracy, after mapping to a hidden X
- Table 7 shows the performance of the Softmax DB on the X token accuracy when teacher-forcing the previous inputs

For Gumbel discrete bottleneck, we present the results in the following tables:

- Table 8 Shows the performance of the Gumbel DB on Autoregressive Z token accuracy, from test inputs
- Table 9 Shows the performance of the Gumbel DB on Autoregressive Z sentence accuracy, from test inputs
- Table 10 shows the performance of the Gumbel DB on the Autoregressive Z reconstruction token accuracy, after mapping to a hidden X
- Table 11 shows the performance of the Gumbel DB on the Autoregressive Z reconstruction sentence accuracy, after mapping to a hidden X
- Table 12 shows the performance of the Gumbel DB on the X token accuracy when teacher-forcing the previous inputs

For VQ discrete bottleneck, we present the results in the following tables:

- Table 13 Shows the performance of the VQ DB on Autoregressive Z token accuracy, from test inputs
- Table 14 Shows the performance of the VQ DB on Autoregressive Z sentence accuracy, from test inputs

Table 3. Softmax DB – Autoregressive Z Token Accuracy. * These baselines are not concerned with the discretizer type and are not trained with our proposed discrete bottleneck. They will appear in all tables for comparison.

SCAN	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)*	–	–	–	–	0.54
T5 Finetuning*	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.78	0.92	0.98	**1.00**	1.00
Joint training	0.76	0.89	0.98	0.99	—
Supervised Pretraining	**0.84**	**0.96**	**0.99**	**1.00**	—
Unsupervised Pretraining	0.79	0.91	0.97	0.99	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.17
T5 Finetuning	0.50	0.74	0.85	0.93	–
Supervised Baseline	0.17	0.30	0.78	0.93	0.97
Joint training	0.56	0.77	**0.94**	0.91	—
Supervised Pretraining	0.47	0.73	0.91	**0.95**	—
Unsupervised Pretraining	**0.58**	**0.82**	0.87	0.91	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.25
T5 Finetuning	0.35	0.72	0.95	0.99	–
Supervised Baseline	0.87	0.94	0.98	0.99	1.00
Joint training	0.94	0.97	**0.99**	**1.00**	—
Supervised Pretraining	0.94	0.93	0.98	**1.00**	—
Unsupervised Pretraining	**0.95**	**0.98**	**0.99**	**1.00**	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.26
T5 Finetuning	0.45	0.63	**0.86**	**0.96**	–
Supervised Baseline	0.62	0.70	0.78	0.82	0.86
Joint training	0.69	0.75	0.84	0.88	—
Supervised Pretraining	**0.73**	**0.80**	0.84	0.88	—
Unsupervised Pretraining	0.71	0.79	0.82	0.85	—

- Table 15 shows the performance of the VQ DB on the Autoregressive Z reconstruction token accuracy, after mapping to a hidden X
- Table 16 shows the performance of the VQ DB on the Autoregressive Z reconstruction sentence accuracy, after mapping to a hidden X
- Table 17 shows the performance of the VQ DB on the X token accuracy when teacher-forcing the previous inputs

Acknowledgments

Robert West's lab is partly supported by grants from Swiss National Science Foundation (200021_185043, TMSGI2_211379), H2020 (952215), Microsoft Swiss Joint Research Center, and by generous gifts from Facebook, Google, and Microsoft. Amin Mansouri was supported in part by the support from Canada CIFAR AI Chair Program, from the Canada Excellence Research Chairs Program, and Microsoft. This research was enabled in part by compute resources provided by Mila, Quebec AI Institute (mila.quebec).

Table 4. Softmax DB – Autoregressive Z Sentence Accuracy

SCAN	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.18	0.52	0.87	0.97	1.00
Joint training	0.24	0.50	0.87	0.95	—
Supervised Pretraining	**0.29**	**0.71**	**0.91**	**0.98**	—
Unsupervised Pretraining	0.25	0.61	0.85	0.95	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.01
T5 Finetuning	0.39	0.61	0.82	0.91	–
Supervised Baseline	0.01	0.11	0.75	0.94	0.97
Joint training	0.45	0.73	**0.94**	0.92	—
Supervised Pretraining	0.32	0.70	0.92	**0.96**	—
Unsupervised Pretraining	**0.47**	**0.81**	0.87	0.91	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.09
T5 Finetuning	0.03	0.45	0.70	0.87	–
Supervised Baseline	0.48	0.71	0.89	0.95	1.00
Joint training	0.63	0.82	**0.95**	**0.97**	—
Supervised Pretraining	0.60	0.83	0.94	**0.97**	—
Unsupervised Pretraining	**0.66**	**0.84**	**0.95**	**0.97**	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.01
T5 Finetuning	0.01	0.13	0.55	0.83	–
Supervised Baseline	0.25	0.40	0.53	0.61	0.69
Joint training	0.43	0.50	0.65	**0.73**	—
Supervised Pretraining	**0.49**	0.59	**0.66**	**0.73**	—
Unsupervised Pretraining	0.47	**0.60**	0.64	0.69	—

Table 5. Softmax DB – Reconstruction Z TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.74	0.81	0.89	0.92	0.96
Joint training	**0.99**	0.98	0.98	**0.97**	—
Supervised Pretraining	**0.99**	**0.99**	**0.99**	**0.97**	—
Unsupervised Pretraining	0.98	0.83	0.93	**0.97**	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.37	0.50	0.74	0.78	0.83
Joint training	0.71	**0.80**	0.86	**0.91**	—
Supervised Pretraining	0.68	0.75	0.86	0.89	—
Unsupervised Pretraining	**0.76**	0.79	**0.88**	0.87	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.92	0.95	0.97	0.98	0.99
Joint training	0.98	**0.99**	**1.00**	**1.00**	—
Supervised Pretraining	0.98	0.97	0.99	**1.00**	—
Unsupervised Pretraining	**0.99**	**0.99**	**1.00**	**1.00**	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.94	0.96	0.97	0.98	0.99
Joint training	0.97	0.97	0.98	**0.99**	—
Supervised Pretraining	**0.98**	0.98	**0.99**	**0.99**	—
Unsupervised Pretraining	**0.98**	**0.99**	**0.99**	**0.99**	—

Table 6. Softmax DB – Reconstruction Z SA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.05	0.11	0.24	0.28	0.46
Joint training	0.82	0.76	0.72	0.65	—
Supervised Pretraining	**0.90**	**0.81**	**0.91**	**0.66**	—
Unsupervised Pretraining	0.84	0.18	0.41	**0.66**	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.01	0.09	0.20	0.28	0.35
Joint training	**0.21**	**0.29**	**0.44**	**0.63**	—
Supervised Pretraining	0.15	0.22	0.33	0.47	—
Unsupervised Pretraining	0.19	0.26	**0.44**	0.41	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.02	0.04	0.35	0.48	0.57
Joint training	0.51	0.76	**0.93**	**0.97**	—
Supervised Pretraining	0.55	0.48	0.68	0.96	—
Unsupervised Pretraining	**0.75**	**0.81**	0.90	0.95	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.11	0.23	0.36	0.48	0.53
Joint training	0.29	0.36	0.50	0.61	—
Supervised Pretraining	0.36	0.44	**0.54**	**0.62**	—
Unsupervised Pretraining	**0.40**	**0.51**	0.52	0.60	—

Table 7. Softmax DB – Teacher-forced X TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	**0.66**	**0.77**	**0.84**	0.88	**0.88**
Joint training	0.57	0.66	0.78	0.84	—
Supervised Pretraining	0.39	0.58	0.70	0.82	—
Unsupervised Pretraining	0.50	0.45	0.69	0.81	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.41	0.50	0.53	0.57	0.65
Joint training	**0.50**	**0.54**	0.57	0.61	—
Supervised Pretraining	0.47	0.50	0.54	0.57	—
Unsupervised Pretraining	0.48	0.50	**0.61**	**0.63**	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.87	0.95	0.98	0.99	1.00
Joint training	**0.90**	**0.96**	0.99	1.00	—
Supervised Pretraining	0.00	0.88	0.95	**0.99**	—
Unsupervised Pretraining	0.88	**0.96**	0.98	0.99	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.74	0.79	0.82	0.85	0.88
Joint training	**0.77**	**0.80**	**0.83**	**0.85**	—
Supervised Pretraining	0.73	**0.80**	**0.83**	**0.85**	—
Unsupervised Pretraining	0.71	0.78	0.82	0.84	—

Table 8. Gumbel DB – Autoregressive Z Token Accuracy

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.75	0.89	0.95	**0.97**	0.97
Joint training	0.76	0.88	0.95	0.96	—
Supervised Pretraining	**0.80**	**0.93**	**0.96**	**0.97**	—
Unsupervised Pretraining	0.79	0.90	**0.96**	0.96	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	**0.91**	–
Supervised Baseline	0.16	0.25	0.70	0.82	0.89
Joint training	0.25	**0.66**	0.81	0.86	—
Supervised Pretraining	0.44	0.62	**0.85**	0.88	—
Unsupervised Pretraining	**0.56**	0.64	0.81	0.87	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	**0.91**	–
Supervised Baseline	0.74	0.80	**0.84**	0.85	0.86
Joint training	0.76	0.79	0.82	0.84	—
Supervised Pretraining	**0.78**	**0.81**	**0.84**	0.84	—
Unsupervised Pretraining	0.75	**0.81**	0.83	0.84	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	**0.69**	**0.91**	–
Supervised Baseline	0.52	0.57	0.63	0.65	0.65
Joint training	0.56	0.62	0.65	0.69	—
Supervised Pretraining	0.55	0.61	0.65	0.68	—
Unsupervised Pretraining	**0.60**	**0.66**	0.68	0.71	—

Table 9. Gumbel DB – Autoregressive Z Sentence Accuracy

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.13	0.46	0.79	**0.86**	0.89
Joint training	0.22	0.48	0.78	0.85	—
Supervised Pretraining	**0.23**	**0.56**	0.80	**0.86**	—
Unsupervised Pretraining	0.21	0.55	**0.81**	**0.86**	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.01	0.08	0.62	0.76	0.84
Joint training	0.06	**0.57**	0.74	0.80	—
Supervised Pretraining	0.26	0.51	**0.78**	**0.81**	—
Unsupervised Pretraining	**0.45**	0.55	0.75	**0.81**	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.30	0.48	0.62	**0.70**	0.73
Joint training	**0.41**	**0.56**	**0.64**	0.68	—
Supervised Pretraining	0.38	0.53	0.63	0.66	—
Unsupervised Pretraining	0.32	0.53	0.63	0.66	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	**0.53**	–
Supervised Baseline	0.14	0.23	0.30	0.33	0.34
Joint training	0.21	0.29	0.34	0.40	—
Supervised Pretraining	0.21	0.29	0.35	0.39	—
Unsupervised Pretraining	**0.29**	**0.37**	0.41	**0.45**	—

Table 10. Gumbel DB – Reconstruction Z TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.74	0.78	0.86	0.90	0.94
Joint training	0.96	0.94	0.96	0.95	—
Supervised Pretraining	0.97	0.98	0.97	0.97	—
Unsupervised Pretraining	0.81	0.88	0.90	0.94	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.33	0.46	0.70	0.75	0.79
Joint training	0.32	0.58	0.73	0.83	—
Supervised Pretraining	0.56	0.63	0.72	0.81	—
Unsupervised Pretraining	0.57	0.75	0.82	0.85	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.90	0.93	0.96	0.97	0.98
Joint training	0.96	0.98	0.98	0.99	—
Supervised Pretraining	0.96	0.97	0.98	0.99	—
Unsupervised Pretraining	0.97	0.98	0.99	0.99	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.92	0.94	0.95	0.96	0.97
Joint training	0.95	0.96	0.96	0.97	—
Supervised Pretraining	0.94	0.95	0.96	0.97	—
Unsupervised Pretraining	0.98	0.98	0.98	0.98	—

Table 11. Gumbel DB – Reconstruction Z SA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.05	0.08	0.17	0.27	0.30
Joint training	0.60	0.43	0.54	0.54	—
Supervised Pretraining	0.65	0.71	0.68	0.62	—
Unsupervised Pretraining	0.11	0.18	0.39	0.55	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.01	0.06	0.20	0.25	0.21
Joint training	0.01	0.11	0.19	0.38	—
Supervised Pretraining	0.07	0.11	0.15	0.25	—
Unsupervised Pretraining	0.05	0.21	0.32	0.42	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.02	0.02	0.20	0.27	0.34
Joint training	0.29	0.48	0.59	0.61	—
Supervised Pretraining	0.24	0.42	0.56	0.60	—
Unsupervised Pretraining	0.30	0.46	0.59	0.63	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.05	0.10	0.17	0.21	0.21
Joint training	0.10	0.14	0.21	0.25	—
Supervised Pretraining	0.07	0.13	0.19	0.27	—
Unsupervised Pretraining	0.26	0.27	0.31	0.34	—

Table 12. Gumbel DB – Teacher-forced X TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.66	0.76	0.84	0.87	0.88
Joint training	0.60	0.70	0.78	0.85	—
Supervised Pretraining	0.40	0.63	0.76	0.84	—
Unsupervised Pretraining	0.36	0.64	0.62	0.76	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.39	0.48	0.51	0.55	0.57
Joint training	0.38	0.49	0.55	0.58	—
Supervised Pretraining	0.45	0.50	0.52	0.56	—
Unsupervised Pretraining	0.43	0.53	0.55	0.57	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.84	0.93	0.97	0.98	0.99
Joint training	0.88	0.95	0.98	0.99	—
Supervised Pretraining	0.86	0.93	0.97	0.98	—
Unsupervised Pretraining	0.85	0.94	0.98	0.99	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.71	0.77	0.80	0.83	0.85
Joint training	0.75	0.79	0.81	0.84	—
Supervised Pretraining	0.72	0.78	0.81	0.84	—
Unsupervised Pretraining	0.69	0.77	0.81	0.84	—

Table 13. VQ DB – Autoregressive Z Token Accuracy

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.73	0.89	0.98	1.00	1.00
Joint training	0.67	0.88	0.95	0.97	—
Supervised Pretraining	0.84	0.95	0.99	1.00	—
Unsupervised Pretraining	0.75	0.91	0.97	0.99	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.23	0.86	0.87	0.93	0.93
Joint training	0.31	0.66	0.90	0.89	—
Supervised Pretraining	0.41	0.86	0.90	0.93	—
Unsupervised Pretraining	0.12	0.13	0.13	0.13	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.86	0.95	0.99	1.00	1.00
Joint training	0.92	0.97	0.99	0.99	—
Supervised Pretraining	0.91	0.97	0.99	1.00	—
Unsupervised Pretraining	0.94	0.97	0.98	0.98	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.54
T5 Finetuning	0.0	0.47	0.69	0.91	–
Supervised Baseline	0.49	0.77	0.91	0.95	0.84
Joint training	0.71	0.81	0.91	0.96	—
Supervised Pretraining	0.69	0.83	0.94	0.96	—
Unsupervised Pretraining	0.44	0.54	0.52	0.61	—

Table 14. VQ DB – Autoregressive Z Sentence Accuracy

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.10	0.41	0.85	0.98	1.00
Joint training	0.12	0.44	0.70	0.87	—
Supervised Pretraining	0.29	0.64	0.91	0.98	—
Unsupervised Pretraining	0.15	0.54	0.84	0.93	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.04	0.86	0.87	0.94	0.91
Joint training	0.14	0.59	0.91	0.89	—
Supervised Pretraining	0.23	0.86	0.90	0.94	—
Unsupervised Pretraining	0.00	0.00	0.00	0.00	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.47	0.74	0.92	0.97	0.84
Joint training	0.51	0.81	0.93	0.97	—
Supervised Pretraining	0.59	0.82	0.94	0.97	—
Unsupervised Pretraining	0.51	0.63	0.62	0.59	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
ICL (GPT3.5)	–	–	–	–	0.11
T5 Finetuning	0.0	0.02	0.08	0.53	–
Supervised Baseline	0.01	0.53	0.79	0.87	0.25
Joint training	0.69	0.78	0.92	0.96	—
Supervised Pretraining	0.41	0.66	0.85	0.91	—
Unsupervised Pretraining	0.01	0.02	0.07	0.05	—

Table 15. VQ DB – Reconstruction Z TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.73	0.77	0.83	0.91	0.99
Joint training	0.95	0.95	0.95	0.96	—
Supervised Pretraining	0.99	0.98	0.98	0.97	—
Unsupervised Pretraining	0.86	0.97	0.99	0.99	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.35	0.67	0.77	0.81	0.53
Joint training	0.59	0.77	0.91	0.93	—
Supervised Pretraining	0.68	0.81	0.83	0.89	—
Unsupervised Pretraining	0.32	0.38	0.32	0.32	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.93	0.98	0.99	1.00	0.98
Joint training	0.97	0.99	0.99	1.00	—
Supervised Pretraining	0.97	0.98	0.99	1.00	—
Unsupervised Pretraining	0.98	0.99	0.98	0.98	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.89	0.95	0.98	0.98	0.98
Joint training	0.97	0.98	0.99	0.99	—
Supervised Pretraining	0.97	0.98	0.99	0.99	—
Unsupervised Pretraining	0.94	0.94	0.94	0.94	—

Table 16. VQ DB – Reconstruction Z SA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.05	0.12	0.29	0.61	0.91
Joint training	0.40	0.38	0.39	0.52	—
Supervised Pretraining	0.89	0.76	0.71	0.66	—
Unsupervised Pretraining	0.17	0.67	0.90	0.85	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.02	0.18	0.19	0.25	0.12
Joint training	0.10	0.30	0.64	0.75	—
Supervised Pretraining	0.13	0.33	0.39	0.56	—
Unsupervised Pretraining	0.00	0.00	0.00	0.00	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.32	0.68	0.78	0.95	0.39
Joint training	0.41	0.68	0.86	0.93	—
Supervised Pretraining	0.35	0.69	0.88	0.96	—
Unsupervised Pretraining	0.44	0.57	0.52	0.53	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.01	0.19	0.51	0.52	0.22
Joint training	0.27	0.40	0.57	0.67	—
Supervised Pretraining	0.19	0.41	0.56	0.71	—
Unsupervised Pretraining	0.00	0.01	0.00	0.00	—

Table 17. VQ DB – Teacher-forced X TA

SCAN	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.65	0.73	0.81	0.86	0.88
Joint training	0.56	0.67	0.74	0.84	—
Supervised Pretraining	0.41	0.62	0.71	0.78	—
Unsupervised Pretraining	0.27	0.61	0.57	0.79	—
PCFG	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.32$	$\eta = 0.99$
Supervised Baseline	0.32	0.52	0.55	0.58	0.62
Joint training	0.43	0.54	0.59	0.65	—
Supervised Pretraining	0.48	0.53	0.55	0.58	—
Unsupervised Pretraining	0.33	0.34	0.33	0.33	—
COGS	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.16$	$\eta = 0.99$
Supervised Baseline	0.85	0.96	0.99	0.99	0.97
Joint training	0.88	0.95	0.98	0.99	—
Supervised Pretraining	0.85	0.93	0.98	0.99	—
Unsupervised Pretraining	0.88	0.93	0.93	0.93	—
CFQ	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.04$	$\eta = 0.08$	$\eta = 0.99$
Supervised Baseline	0.60	0.77	0.83	0.86	0.88
Joint training	0.74	0.78	0.83	0.86	—
Supervised Pretraining	0.68	0.78	0.84	0.86	—
Unsupervised Pretraining	0.49	0.55	0.53	0.53	—