Challenging the Semi-Supervised VAE Framework for Text Classification

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

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

Abstract

Semi-Supervised Variational Autoencoders (SSVAEs) are widely used models for data efficient learning. In this paper, we question the adequacy of the standard design of sequence SSVAEs for the task of text classification as we exhibit two sources of overcomplexity for which we provide simplifications.

These simplifications to SSVAEs preserve their theoretical soundness while providing a number of practical advantages in the semi-supervised setup where the result of training is a text classifier. These simplifications are the removal of *(i)* the Kullback-Liebler divergence from its objective and *(ii)* the fully unobserved latent variable from its probabilistic model. These changes relieve users from choosing a prior for their latent variables, make the model smaller and faster, and allow for a better flow of information into the latent variables.

We compare the simplified versions to standard SSVAEs on 4 text classification tasks. On top of the above-mentioned simplification, experiments show a speed-up of 26%, while keeping equivalent classification scores. The code to reproduce our experiments is public[1].

1 Introduction

Obtaining labeled data to train NLP systems is a process that has often proven to be costly and time-consuming, and this is still largely the case (Martínez Alonso et al., 2016; Seddah et al., 2020). Consequently, semi-supervised approaches are appealing to improve performance while alleviating dependence on annotations. To that end, Variational Autoencoders (VAEs) (Kingma and Welling, 2014) have been adapted to semi-supervised learning (Kingma et al., 2014), and subsequently applied to several NLP tasks (Chen et al., 2018a; Corro and Titov, 2019; Gururangan et al., 2020).

A notable difference between the generative model case from where VAEs originate, and the semi-supervised case is that only the decoder (generator) of the VAE is kept after training in the first case, while in the second, it is the encoder (classifier) that we keep. This difference, as well as the autoregressive nature of text generators has not sufficiently been taken into account in the adaptation of VAEs to semi-supervised text classification. In this work, we show that some components can be ablated from the long used semi-supervised VAEs (SSVAEs) when only aiming for text classification. These ablations simplify SSVAEs and offer several practical advantages while preserving their performance and theoretical soundness.

The usage of unlabeled data through SSVAEs is often described as a *regularization* on representations (Chen et al., 2018a; Wolf-Sonkin et al., 2018; Yacoby et al., 2020). More specifically, SSVAEs add to the supervised learning signal, a conditional generation learning signal that is used to train on unlabeled samples. From this observation, we study two changes to the standard SSVAE framework. The first simplification we study is the removal of a term from the objective of SSVAEs: the Kullback-Leibler term. This encourages the flow of information into latent variables, frees the users from choosing priors for their latent variables, and is harmless to the theoretical soundness of the semi-supervised framework. The second simplification we study is made to account for the autoregressive nature of text generators. In the general case, input samples in SSVAEs are described with two latent variables: a partially-observed latent variable, which is also used to infer the label for the supervised learning task, and an unobserved latent variable, which describes the rest of the variability in the data. However, autoregressive text generators are powerful enough to converge without the need for latent variables. Therefore, removing the unobserved latent variable is the second change we study in SSVAEs. The above modifications can be found in some rare works throughout the literature,

[1] https://github.com/ghazi-f/Challenging-SSVAEs

e.g. (Corro and Titov, 2019). We, however, aim to provide justification for these changes beyond the empirical gains that they exhibit for some tasks.

Our experiments on four text classification datasets show no harm to the empirical classification performance of SSVAE in applying the simplifications above. Additionally, we show that removing the unobserved latent variable leads to a significant speed-up.

To summarize our contribution, we justify two simplifications to the standard SSVAE framework, explain the practical advantage of applying these modifications, and provide empirical results showing that they speed up the training process while causing no harm to the classification performance.

2 Background

2.1 Variational Autoencoders

Variational Autoencoders (Kingma and Welling, 2019) are a class of generative models that combine Variational Inference with Deep Learning modules to train a generative model. For a latent variable z, and an observed variable x, the generative model p_θ consists of a prior $p_\theta(z)$ and a decoder $p_\theta(x|z)$. VAEs also include an approximate posterior (also called the encoder) $q_\phi(z|x)$. Both are used during training to maximize an objective called the Evidence Lower Bound (ELBo), a lower-bound of the log-likelihood:

$$\begin{aligned}\log p_\theta(x) \geq \\ \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - \mathrm{KL}\left[q_\phi(z|x); p_\theta(z)\right] \\ = \mathrm{ELBo}(x; z) \qquad (1)\end{aligned}$$

Throughout the paper, we will continue to use this $\mathrm{ELBo}(.;.)$ operator, with the observed variable(s) as a first argument, and the latent variable(s) as a second argument. In the original VAE framework, after training, the encoder q_ϕ is discarded and only the generative model (the prior and the decoder) are kept.

2.2 Semi-Supervised VAEs

The idea of using the VAE encoder as a classifier for semi-supervised learning has first been explored in (Kingma et al., 2014). Besides the usual unobserved latent variable z, the semi-supervised VAE framework also uses a partially-observed latent variable y. The encoder $q_\phi(y|x)$ serves both as the inference module for the supervised task, and as an approximate posterior (and encoder) for the y variable in the VAE framework.

Consider a set of labeled examples $L = \{(x_1, y_1), ..., (x_{|L|}, y_{|L|})\}$, and a set of unlabeled examples $U = \{x'_1, ..., x'_{|U|}\}$. For the set L, $q_\phi(y|x)$ is trained *i)* with the usual supervised objective (typically, a cross-entropy objective for a classification task) *ii)* with an ELBo that considers x and y to be *observed*, and z to be a *latent* variable. A weight α is used on the supervised objective to control its balance with ELBo. For the set U, $q_\phi(y|x)$ is only trained as part of the VAE model with an ELBO where y is used, this time, as a *latent* variable like z. Formally, the training objective $\mathcal{J}^\alpha$ of a SSVAE is as follows:

$$\begin{aligned}\mathcal{J}^\alpha = \sum_{(x,y)\in L} \Big(\mathrm{ELBo}((x, y); z) + \alpha \ \log q_\phi(y|x)\Big) \\ + \sum_{x\in U} \mathrm{ELBo}(x; (y, z)) \qquad (2)\end{aligned}$$

3 Simplifying SSVAEs for Text Classification

The simplifications we propose stem from the analysis of an alternative form under which ELBO can be written (Eq. 2.8 in Kingma and Welling, 2019). Although it is valid for any arguments of $\mathrm{ELBo}(.;.)$, we display it here for an observed variable x, and the couple of latent variables (y, z):

$$\begin{aligned}\mathrm{ELBo}(x; (y, z)) = \\ \log p_\theta(x) - \mathrm{KL}[q_\phi(y, z|x)||p_\theta(y, z|x)] \qquad (3)\end{aligned}$$

For the case of SSVAEs, this form provides a clear reading of the additional effect of ELBo on the learning process: *i)* maximizing the log-likelihood of the generative model $p_\theta(x)$, *ii)* bringing the parameters of the inference model $q_\phi(y, z|x)$ closer to the posterior of the generative model $p_\theta(y, z|x)$. Since $p_\theta(y, z|x)$ is the distribution of the latent variables expected by the generative model p_θ for it to be able to generate x, we can conclude that ELBo trains *both* latent variables for conditional generation on the unsupervised dataset U.

3.1 Dropping the Unobserved Latent Variable

Building on observations from equation 3, we question the usefulness of training both latent variables for conditional generation when semi-supervised learning only aims for an improvement on the inference of the partially-observed latent variable y.

For the case of language generation, the sequence of discrete symbols in each sample is often modeled by an autoregressive distribution $p_\theta(x|y,z) = \prod_i p_\theta(x_i|y,z,x_{<i})$ where x_i is the i^{th} symbol in the sequence, and $x_{<i}$ are the symbols preceding x_i. Such a distribution is able to generate realistic samples when trained on a target text corpus, so much that text VAEs are plagued with a problem known as *posterior collapse* (Bowman et al., 2016) where the latent variable is ignored by the generative model. We therefore propose to keep only y and to drop z from the model avoiding its presence in the Kullback-Leibler divergence in Equation 3 and saving some parameters.[2]

3.2 Dropping the Kullback-Leibler Term

Previous work on VAE-based language models showed that the KL divergence in Eq. 1 sometimes discourages the model from using latent variables and makes them useless in practice (Bowman et al., 2016; Zhao et al., 2017; Chen et al., 2018b).

An interesting result from Zhao et al. (2017) is that ELBo without KL divergence (*KL-free*) is still a theoretically sound objective for generative modeling with VAEs. The difference between the generative model resulting from a regular ELBo and a KL-free ELBo is the prior of the model. A KL-free ELBo results in a generative model that uses as a prior $q_\phi(z) = \int_z q_\phi(z|x)p_{data}(x)dx$. This prior is intractable which makes the resulting model impractical for generation, but causes no problem for semi-supervised VAEs. We therefore propose, as a second change to the standard SSVAE framework, the removal of the KL-divergence in Eq. 1.

Note that in this case, the network formulates its own prior instead of requiring the user to choose it. That is a significant advantage since the choice of a good prior is difficult: it must model adequately the *default* behavior of the latent variables, and requires a closed form for the KL-divergence in Eq. 1 to stabilize training.

3.3 Resulting Objective

Applying both of the previous simplifications to the semi-supervised objective in Eq. 2 leads to the following objective:

$$\sum_{(x,y)\in L} \Big(\log p_\theta(x|y) + \alpha \ \log q_\phi(y|x) \Big) \\ + \sum_{x\in U} \mathbb{E}_{y\sim q_\phi(y|x)} \left[\log p_\theta(x|y)\right]) \qquad (4)$$

As can be seen, the first ELBo in Eq. 2 turns into a supervised conditional generation objective, while the second ELBo turns into a reconstruction term that relies only on y. Nevertheless, we stress that the second term is still an ELBo, and the whole objective is still a VAE-based semi-supervised learning objective. It should also be noted that, without z, the latent variables cannot provide the decoder with the full information about a sentence and, therefore, cannot reach a state where each sample is *reconstructed*. To avoid confusion, instead of *reconstructing* from y, the role of the reconstruction term is better read in our case as *raising the probability* of the sample at hand under the associated label y.

4 Experiments

In this section, we display comparisons between instances of standard SSVAEs and the same SSVAEs after applying the changes we propose.

4.1 Setup

Datasets We consider 4 Datasets for our study: the IMDB (Maas et al., 2011) and Yelp review (Li et al., 2018) binary sentiment analysis datasets, and the AG News and DBPedia (Zhang et al., 2015) topic classification datasets. The Datasets have been chosen to represent a range over different tasks (Sentiment Analysis and Topic Classification), different numbers of classes, and different sentence lengths. A summary of dataset statistics is in Table 1.

dataset	Labels	Av. Sample length	N° Classes
AG News	Topic	37.85±10.09	4
DBPedia	Topic	46.13±22.46	14
IMDB	Sentiment	233.79±173.72	2
Yelp	Sentiment	8.88±3.64	2

Table 1: Dataset properties.

As was done in Chen et al. (2020), we measure performance on the different datasets with equal numbers of samples. Accordingly, for each dataset, we randomly subsample 10K samples from the

[2] In this case, one may be tempted to drop the VAE framework entirely and resort to other learning algorithms such as EM or direct likelihood maximization. Although possible in theory, this would disconnect q_ϕ from the generator's training, and thus discard the benefit from using unlabeled data.

Objective	AGNEWS	DBPedia	IMDB	Yelp
Supervised	86.98(0.74)	96.97(0.28)	81.02(0.64)	92.47(0.48)
SSVAE	87.89(0.54)	**97.75**(0.11)	83.34(0.91)	92.85(0.78)
SSVAE-{KL}	**87.95**(0.19)	97.58(0.13)	83.87(0.47)	92.90(0.54)
SSVAE-{z}	87.94(0.33)	97.40(0.14)	81.90(5.17)	93.60(0.74)
SSVAE-{KL, z}	87.85(0.29)	97.58(0.19)	**84.79**(1.34)	**93.77**(0.61)

Table 2: Accuracies On AGNEWS, DBPedia, IMDB, and Yelp. The values are averages over 5 runs with standard deviations between parentheses. The best score for each dataset and each amount of labeled data is given in bold.

original training set as *unlabeled* data. We also use 4K labeled samples a training set and 1K as development set. We use the original test sets from each dataset. All the samples are tokenized using a simple whitespace tokenizer.

Network Architecture The size of z is set to 32. For experiments without z, we simply drop all the components associated to it from the network.

The encoder consists of a pre-trained 300-dimensional fastText (Bojanowski et al., 2017) embedding layer, and 2 Bidirectional LSTM networks with 100 hidden states each, one for each of the latent variables y and z. The logits of y are then obtained by passing the last state of its Bidirectional LSTM through a linear layer. Similarly the last state of the Bidirectional LSTM for z is passed through a linear layer to obtain its mean parameter, and a linear layer with a softplus activation to obtain its standard deviation parameter.

As for the decoding step, to allow backpropagation, z is sampled using the reparameterization trick (Kingma and Welling, 2014), and y is sampled using the Gumbel-Softmax trick (Jang et al., 2017). Xu et al. (2017) have shown that latent variables are best exploited in SSVAEs when concatenated with the previous word at each generation step to obtain the next word. We design our decoder accordingly and use a 1-layered LSTM with size 200. The only hyper-parameter we tune on the development set is α, the coefficient weighting the supervised learning objective in Eq. 2, which is selected in the set $\{10^0, 10^{-1}, 10^{-2}, 10^{-3}\}$. Further implementation details are provided in Appendix A.

4.2 Results

Classification performance In Table 2, we compare the performance of a standard SSVAE, to a SSVAE where we remove the KL-divergence (SSVAE-{KL}) another where z is removed (SSVAE-{z}) and a third version where both the KL-divergence and z are removed (SSVAE-{KL, z}). We measure performance on all datasets using *accuracy*. As a baseline, we also include the results of an objective that does not use unlabeled data. The architecture we use for this objective is simply the LSTM encoder that we use to obtain y for the SSVAE objectives. This baseline is referred to as *Supervised*.

The aim of our experiment is to see whether we observe that there is a harm to the performance of SSVAEs when applying the proposed simplifications. In Table 2, we see that applying both changes compares favorably to the standard SSVAE 2 times out of 4. The removal of z yields the same comparison, while removing the KL term causes improvement 3 times out of 4. For more extensive testing, we ran experiments for varying amounts of labeled data (from 1% to 100%; *cf.* Appendix C), and only found 4 statistically significant differences between SSVAE and its variants: 3 in favor of one of our Simplified SSVAEs, and 1 in favor of the standard SSVAE.

We performed additional experiments in an out-of-domain setting (*c.f* Appendix B.) using our sentiment analysis datasets, and also observed improvements with our simplifications.

Speeding Up the Learning Process By removing the KL-divergence and the components associated with z, an improvement on the speed of the learning process is to be expected. This improvement is highly dependent on the model and on the implementation at hand. As an example, we measure the average speed of an optimization iteration for each dataset, and each version of SSVAE. In Table 3, the speed of each objective is displayed proportionally to the speed of standard SSVAEs. The calculations associated with the KL-divergence do not seem to slow down the iterations. However, removing z and its associated components consistently cuts out a considerable proportion of the duration of optimization steps. This proportion ranges from 14% (DBPedia) to 26%(AGNEWS).

Dataset	SSVAE-{KL}	SSVAE-{z}	SSVAE-{KL, z}
AGNEWS	0.911(0.73)	**0.742**(0.65)	**0.742**(0.81)
DBPedia	1.03(0.56)	**0.861**(0.61)	0.867(0.56)
IMDB	1.018(0.25)	0.822(0.25)	**0.816**(0.25)
Yelp	0.986(1.52)	**0.819**(1.39)	**0.819**(1.52)

Table 3: Training durations for each objective relative to standard SSVAE, averaged over 200 iterations. Standard deviations are given between parentheses. Lowest duration for each dataset is given in bold.

5 Related Works

After the pioneering work of Kingma et al. (2014), SSVAEs were extended to tasks such as morphological inflections (Wolf-Sonkin et al., 2018), controllable speech synthesis (Habib et al., 2019), parsing (Corro and Titov, 2019), sequential labeling (Chen et al., 2018a) among many others. VAE internals have also been *tweaked* in various manners to improve the learning performance. For instance, Gururangan et al. (2020) introduce a low resource pretraining scheme to improve transfer with VAEs, while Zhang et al. (2019) propose to use the deterministic ancestor of a latent variable to perform classification, and constrain it with an adversarial term to have it abide by the values of the random latent variable.

While our work is a focused contribution dedicated to the theoretical soundness and the practical advantages of two simplifications to the SSVAE framework for text classifications, it could be extendend to other tasks involving text generation as the unsupervised VAE objective. For instance, the work of Corro and Titov (2019) shows that semi-supervised dependency parsing scores higher with both the changes we study.

6 Conclusion

Starting from the observation that SSVAEs can be viewed as the combination of a supervised learning signal with an unsupervised conditional generation learning signal, we show that this framework needs neither to include a KL-divergence nor an unobserved latent variable (z) when dealing with text classification. We subsequently perform experimental comparisons between standard SSVAEs and simplified SSVAEs that indicate that they are globally equivalent in performance.

Our changes provide a number of practical advantages. First, removing the KL-divergence frees practitioners from choosing priors for the variables they use, and allows information to flow freely into these variables. Second, removing the latent variable z from the computational graph speeds up computation and shrinks the size of the network. Despite their popularity, VAEs are often tedious to train for NLP tasks. In that regard, our simplifications should facilite their usage in future works.

Acknowledgments

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

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

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

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018a. Variational sequential labelers for semi-supervised learning. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 215–226.

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

Caio Corro and Ivan Titov. 2019. Differentiable perturb-and-parse: Semi-supervised parsing with a structured variational autoencoder. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–24.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A Smith. 2020. Variational pretraining for semi-supervised text classification. *ACL 2019 - 57th*

Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, pages 5880–5894.

Raza Habib, Soroosh Mariooryad, Matt Shannon, Eric Battenberg, R J Skerry-Ryan, Daisy Stanton, David Kao, and Tom Bagby. 2019. Semi-Supervised Generative Modeling for Controllable Speech Synthesis. In *International Conference on Learning Representations*, pages 1–13.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–13.

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

Diederik P Kingma, Danilo J Rezende, Shakir Mohamed, and Max Welling. 2014. Semi-Supervised Learning with Deep Generative Models. *Advances in Neural Information Processing Systems*, 4(January):3581–3589.

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

Diederik P Kingma and Max Welling. 2019. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392.

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

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1:142–150.

Héctor Martínez Alonso, Djamé Seddah, and Benoît Sagot. 2016. From noisy questions to Minecraft texts: Annotation challenges in extreme syntax scenario. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 13–23, Osaka, Japan. The COLING 2016 Organizing Committee.

Geoffrey Roeder, Yuhuai Wu, and David Duvenaud. 2017. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. *Advances in Neural Information Processing Systems*, 2017-Decem:6926–6935.

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

Lawrence Wolf-Sonkin, Jason Naradowsky, Sebastian J Mielke, and Ryan Cotterell. 2018. A structured variational autoencoder for contextual morphological inflection. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:2631–2641.

Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 3358–3364.

Yaniv Yacoby, Weiwei Pan, and Finale Doshi-Velez. 2020. Failure Modes of Variational Autoencoders and Their Effects on Downstream Tasks. In *ICML Workshop on Uncertainty {\&} Robustness in Deep Learning (UDL)*.

Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. 2019. Advances in Variational Inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):2008–2026.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. 2017. Towards deeper understanding of variational autoencoding models. *CoRR*, abs/1702.08658.

Dataset	Supervised	SSVAE	SSVAE-{KL}	SSVAE-{z}	SSVAE-{KL, z}
IMDB⟶Yelp	59.07(1.19)	61.78(6.03)	68.67$^+$(4.85)	**71.30**$^+$(7.67)	64.69$^+$(3.84)
Yelp⟶IMDB	66.17(2.62)	**69.54**(2.49)	66.67(3.26)	65.15(2.31)	66.13(3.82)

Table 4: Out-of-domain Accuracies between IMDB and Yelp for the different objectives. The best objective for each out-of-domain inference direction is given in bold. The scores displaying statistically significant improvement compared to the score of the supervised objective are marked with $^+$

A Implementation Details

Training and validation data splits We sample 5 *labeled* data splits of size 1K. Each of these 5 splits will, in turn, play the role of validation set for one experiment, while the other 4 splits are used for training. Looping over these splits yields 5 runs for each experiment. The results we display are the average (and standard deviation) of the results for each of these runs. The validation score serves selecting hyper-parameters (in our case only α from Eq. 2). The final test scores are measured on the original test set of each dataset.

Probabilistic Graphical Model For models that use both z and y, we consider the latent variables to be conditionally independent in the inference model (*i.e.* $q_\phi(y, z|x) = q_\phi(y|x)q_\phi(z|x)$)) and independent in the generation model (*i.e* $p_\theta(y, z) = p(y)p(z)$).

Training Procedure We use the STL estimator (Roeder et al., 2017) which is a low-variance unbiased gradient estimator for ELBo.

The network is optimized using ADAM (Kingma and Ba, 2015), with a learning rate of 4e-3 and a dropout rate of 0.5. If the accuracy on the validation set doesn't increase for 4 epochs, the learning rate is divided by 4. If it doesn't increase for 8 epochs, the training is stopped. For objectives that include a KL-divergence, we scale it with a coefficient that is null for 3K steps then linearly increased to 1 for the following 3K steps to avoid posterior collapse (Li et al., 2020).

B Out-of-domain experiments

The sentiment analysis tasks we use for these experiments take place in different domains (Restaurant reviews for Yelp, and Movie reviews for IMDB). Using models trained for each domain (with %100 of the data), we measure performance on the other domain to see whether the changes we study have an effect on out-of-domain generalization. In Table 4, we compare the out-of-domain performances of each of the objectives to that of the baseline that doesn't use unlabeled data (*Supervised*).

The table shows no statistically significant gains from using unlabeled Yelp training data for inference on IMDB. This is to be expected as reviews from Yelp are drastically shorter than those from IMDB (*cf.* Table 1). However, for out-of-domain inference in the opposite direction, all the semi-supervised objectives except the standard SSVAE show statistically significant gains. Removing the KL-divergence to accumulate more information in y, and removing z to have conditional generation exclusively rely on y seem to be effective to help generalization beyond the original domain of the task.

C Results Over Varying Amounts of Data

We display results with varying amounts of data in Table 5.

Dataset	Objective	1%	3%	10%	30%	100%
IMDB	Supervised	54.62(3.30)	56.47(1.02)	62.01(2.75)	69.65(2.02)	81.02(0.64)
	SSVAE	53.92(2.34)	56.03(4.20)	62.15(5.03)	75.39(0.49)	83.34(0.91)
	SSVAE-{KL}	52.70(1.72)	54.95(0.77)	62.37(4.45)	74.18(1.97)	83.87(0.47)
	SSVAE-{z}	**54.15**(2.46)	**56.86**(1.77)	62.15(2.87)	75.42(1.80)	81.90(5.17)
	SSVAE-{KL, z}	53.51(1.99)	56.58(2.22)	**63.24**(4.15)	**75.87**(1.30)	**84.79**(1.34)
AGNEWS	Supervised	68.60(4.88)	75.92(1.74)	81.96(0.83)	84.59(0.67)	86.98(0.74)
	SSVAE	65.79(5.02)	75.95(1.27)	82.47(0.43)	85.50(0.30)	87.89(0.54)
	SSVAE-{KL}	**68.56**(1.89)	76.25(2.21)	82.76(0.45)	85.73(0.80)	**87.95**(0.19)
	SSVAE-{z}	67.13(6.55)	**77.28**(1.81)	**83.48***(0.75)	**85.75**(0.74)	87.94(0.33)
	SSVAE-{KL, z}	66.96(3.42)	76.47(1.24)	82.58(0.97)	85.51(0.57)	87.85(0.29)
Yelp	Supervised	70.32(1.84)	76.32(2.07)	83.41(1.75)	87.85(0.58)	92.47(0.48)
	SSVAE	**71.34**(1.93)	76.96(1.64)	82.96(0.69)	89.35(0.39)	92.85(0.78)
	SSVAE-{KL}	69.85(2.86)	76.82(1.31)	82.90(2.23)	88.33(0.99)	92.90(0.54)
	SSVAE-{z}	68.74(2.95)	**78.26**(1.70)	84.11(1.25)	**90.27***(0.28)	93.60(0.74)
	SSVAE-{KL, z}	69.21(1.10)	77.30(2.57)	**85.02***(1.24)	89.74(1.31)	**93.77**(0.61)
DBPedia	Supervised	63.67(1.74)	81.49(2.25)	90.56(1.21)	94.63(0.32)	96.97(0.28)
	SSVAE	64.42(1.83)	83.16(1.49)	92.95(0.82)	96.26(0.25)	**97.75**(0.11)
	SSVAE-{KL}	**66.09**(3.05)	81.97(1.54)	**93.64**(0.76)	96.32(0.28)	97.58(0.13)
	SSVAE-{z}	62.56(5.60)	**83.40**(2.42)	93.37(1.00)	**96.39**(0.21)	97.40†(0.14)
	SSVAE-{KL, z}	62.15(1.68)	82.67(2.16)	93.40(1.10)	96.31(0.24)	97.58(0.19)

Table 5: Accuracies on IMDB, AGNEWS, Yelp and DBPedia with varying amount of **labeled** data. The values are averages over 5 runs with standard deviations between parentheses. The best score for each dataset and each amount of labeled data is given in bold. Each semi-supervised objective that scores above (resp. below) SSVAE with p-value<0.05 is marked with * (resp. †)