Generative semi-supervised learning with a neural seq2seq noisy channel

Soroosh Mariooryad [*1] Matt Shannon [*1]
Siyuan Ma [1] Tom Bagby [1] David Kao [1] Daisy Stanton [1] Eric Battenberg [1] RJ Skerry-Ryan [1]

Abstract

We use a neural noisy channel generative model to learn the relationship between two sequences, for example text and speech, from little paired data. We identify time locality as a key assumption which is restrictive enough to support semi-supervised learning but general enough to be widely applicable. Experimentally we show that our approach is capable of recovering the relationship between written and spoken language (represented as graphemes and phonemes) from only 5 minutes of paired data. Our results pave the way for more widespread adoption of generative semi-supervised learning for seq2seq tasks.

1. INTRODUCTION

Generative modeling, that is modeling the joint probability distribution of all observable quantities, has long held promise as a principled approach to semi-supervised learning (Cooper & Freeman, 1970; Kingma et al., 2014). In this approach, the parameters of the joint distribution are tuned to find the best explanation of all observed data, marginalizing over any quantities that are not observed for a given example. Assumptions about how the observable quantities relate, essential to any approach to semi-supervised learning, are explicitly encoded in the structure of the model. Several issues have limited the widespread applicability generative semi-supervised learning: it is often intractable to compute the required marginal probabilities, it is sensitive to incorrect modeling assumptions, and it can be difficult to identify assumptions which are general enough to approximate reality but restrictive enough to support meaningful learning.

In this paper we apply generative semi-supervised learning to seq2seq tasks, where the goal is to learn the relationship between two sequences. We address intractability by using a variational approximation extending Kingma et al. (2014) to cope with a sequential latent, address sensitivity to incorrect modeling assumptions by using powerful modern neural seq2seq models as components of our joint model, and identify *time locality* of the relationship between the two sequences as a plausible assumption for a range of seq2seq tasks such as automatic speech recognition (ASR), text-to-speech (TTS) synthesis, machine translation, optical character recognition (OCR), and text summarization.

Our approach extends the *noisy channel model* traditionally used for ASR, OCR and machine translation. In this formulation, a sequence x, say text, is drawn from some distribution and statistically transformed into a second sequence y, say speech audio; the speech recognition task is then to invert this generative model to infer the text most likely to have given rise to a given speech waveform. This generative model of speech was historically successful (Baker, 1975; Jelinek, 1976; Rabiner, 1989), but has been superseded in modern discriminative systems by directly modeling the conditional distribution of text given speech (Graves et al., 2006; Amodei et al., 2016), allowing limited modeling power to be solely devoted to the task of interest and of not requiring faulty modeling assumptions when modeling high-dimensional y values for the sake of tractability. However learning from untranscribed speech audio in a principled way is fundamentally impossible with this approach.

We explore a noisy channel joint model of text and speech for learning from a corpus consisting of relatively large amounts of text-only and speech-only data, but little or no parallel (text, speech) data. Our core contributions are:

- Updating the traditional noisy channel model used for speech recognition and synthesis to support a neural LM source and neural seq2seq channel, maintaining efficient inference via a variational approximation (§2).
- Semi-supervised experiments showing that this model can learn the relationship between written and spoken language from only 5 minutes of paired data (§4).
- Discussion of the importance of *assumptions* for learning from little or no paired data (§3 and §6):
  - Emphasizing that learning a joint from little or no paired data always requires making assumptions.

[*]Equal contribution [1]Google, Mountain View, California, USA. Correspondence to: Soroosh Mariooryad <soroosh@google.com>, Matt Shannon <mattshannon@google.com>.

ICML 2023 Workshop on Structured Probabilistic Inference & Generative Modeling, Honolulu, Hawaii, USA. Copyright 2023 by the author(s).

– Arguing that for principled semi-supervised learning it is desirable to make assumptions explicit.

2. MODEL

In this section we describe our approach to generative semi-supervised learning. One of the advantages of generative modeling is that it provides a principled way to use unpaired data during training (Cooper & Freeman, 1970).

We assume some process generates pairs (x, y) of a sequence $x = [x_s]_{s=0}^{S-1}$ and a sequence $y = [y_t]_{t=0}^{T-1}$ not necessarily of the same length. For example x might be text and y a sequence of mel spectrogram frames derived from corresponding speech audio. We assume that we have a dataset consisting of a fraction α of examples where we only observe x, a fraction β of examples where we only observe y, and a fraction γ where we observe both x and y, where $\alpha + \beta + \gamma = 1$. We are interested in the regime where the *paired fraction* γ is small or zero.

We model the process which generates (x, y) pairs using a generative model $p_\lambda(x, y) = p_\lambda(x) p_\lambda(y|x)$.[*] This is a form of *noisy channel model* (Kernighan et al., 1990). We also find it helpful to introduce a *variational posterior* $q_\nu(x|y)$ which approximates $p_\lambda(x|y)$. If x is text and y is speech then $p_\lambda(x)$ is a language model, $p_\lambda(y|x)$ is a speech synthesis model and $q_\nu(x|y)$ is a speech recognition model. We use recurrent neural autoregressive models for $p_\lambda(x)$, $p_\lambda(y|x)$ and $q_\nu(x|y)$. Full details are given in §D.1. We discuss desirable constraints on $p_\lambda(x)$, $p_\lambda(y|x)$ and $q_\nu(x|y)$ to support effective learning from unpaired data in §3.

We estimate the parameters λ of the generative model by approximate maximum likelihood estimation on the available data. On examples where only y is observed, the likelihood is the marginal $p_\lambda(y) = \sum_x p_\lambda(x) p_\lambda(y|x)$, which is intractable for our flexible neural $p_\lambda(x)$. The variational posterior $q_\nu(x|y)$ allows us to approximate this intractable marginal using the wake–sleep algorithm (Hinton et al., 1995). Our final training objective for a single example is

$$l^{\text{gen}} = \begin{cases} \log p_\lambda(x) & \text{if } x \text{ obs} \\ \log p_\lambda(x^{\text{var}}, y) - \log q_\nu(x^{\text{var}}|y) & \text{if } y \text{ obs} \\ \log p_\lambda(x, y) & \text{if both obs} \end{cases} \tag{1}$$

$$l^{\text{var}} = \log q_\nu(x^{\text{gen}}|y^{\text{gen}}) \tag{2}$$

where $x^{\text{var}} \sim q_\nu(x|y)$ and $(x^{\text{gen}}, y^{\text{gen}}) \sim p_\lambda(x, y)$. We learn (λ, ν) by simultaneous gradient ascent based on $(\partial l^{\text{gen}}/\partial\lambda, \partial l^{\text{var}}/\partial\nu)$. The objective l^{var} trains $q_\nu(x|y)$ to approximate $p_\lambda(x|y)$. When this approximation is exact, l^{gen} is equal to the true log likelihood; in general it is a lower bound. Full details are given in §A.

[*]In preliminary experiments this performed better than assuming x and y are each generated from a shared sequential latent variable z, that is $p_\lambda(x, y) = \sum_z p_\lambda(z) p_\lambda(x|z) p_\lambda(y|z)$.

3. TIME LOCALITY

We propose *time locality* as an assumption that is general enough to be applicable to many practical seq2seq tasks but restrictive enough to potentially allow learning from unpaired data. We say a decoder is *strictly time local* if its probability can be written as a product of factors

$$p_\lambda(y|x) = \prod_{t=0}^{T-1} f_t(y_{t-L:t+L}, x_{\overline{s}(t)-K:\overline{s}(t)+K}) \tag{3}$$

for some *time constants* $K, L \in \mathbb{Z}_{\geq 0}$, where $\overline{s} : \{0, \dots, T-1\} \to \{0, \dots, S-1\}$ is a function aligning each position in y to a position in x. For example, the decoder $p_\lambda(y|x) = \prod_t p_\lambda(y_t|x_{\overline{s}(t)})$ used in traditional noisy channel models of speech based on hidden Markov models is strictly time local with $K = L = 0$. We refer to a decoder as time local if (3) holds approximately. If we assume that the true marginal over y has long-range correlations with time constant much greater than L, then the only way for the model as a whole to capture these correlations across time in its marginal $p_\lambda(y)$ is to induce them from corresponding correlations across time in x. This provides the generative model with an incentive to uncover how x maps to y. Time locality is an intuitively reasonable assumption in many seq2seq problems such as speech recognition and synthesis, optical character recognition and machine translation (with non-monotonic $\overline{s}$). We show time locality aids semi-supervised learning empirically in §4 and theoretically in §B, where we show that it allows learning from no paired data in a highly simplified version of our full setup.

4. EXPERIMENTS

We apply the proposed approach to learning the relationship between written and spoken language. We represent spoken language as an audio-derived phoneme sequence and evaluate semi-supervised utterance-level *grapheme-to-phoneme (g2p)* and *phoneme-to-grapheme (p2g)*.

4.1. Experimental setup

Our experimental setup is described in detail in §D.1. We use LibriTTS (Zen et al., 2019). We compute *phoneme error rate (PER)* for a generative model with x a grapheme sequence and y a phoneme sequence, and *character error rate (CER)* for a generative model with x a phoneme sequence and y a grapheme sequence. Based on the discussion in §2 and §3, our guiding principle for choosing the model architectures is to make $p_\lambda(x)$ and $q_\nu(x|y)$ as flexible as possible and to restrict $p_\lambda(y|x)$ to be time local. We use recurrent autoregressive models for all three. We impose time local-

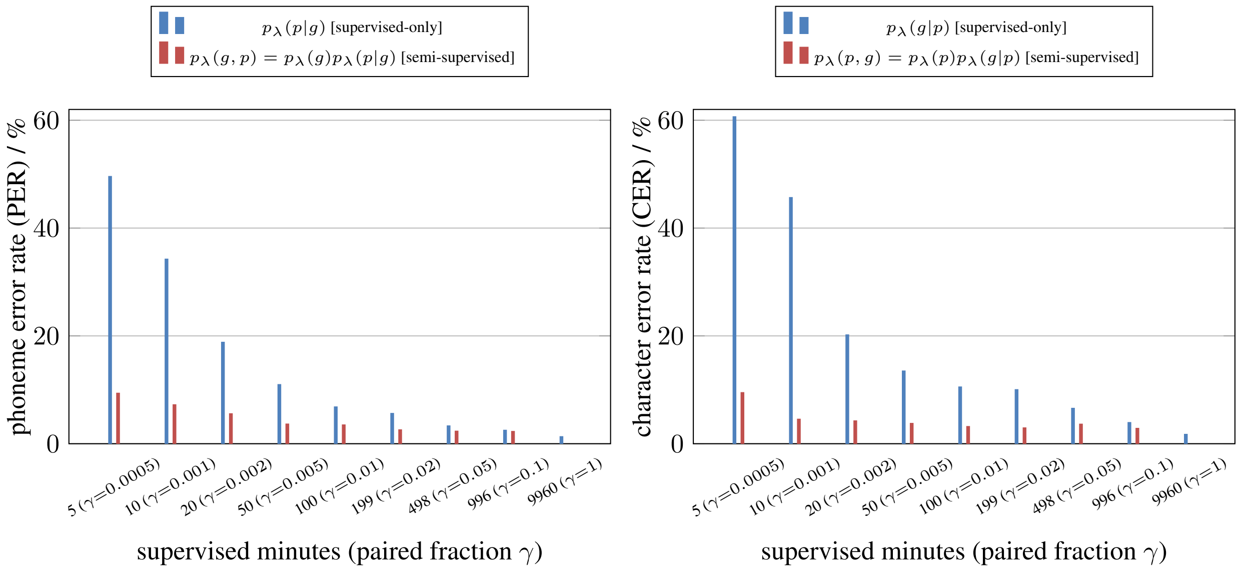

Figure 1. Phoneme (p) and grapheme (g) prediction error rates at varying paired fractions (γ), for a reference supervised-only model (blue) and the proposed generative model consuming both paired and unpaired data (red), on LibriTTS test set.

ity on $p_\lambda(y|x)$ by ensuring that $p_\lambda(y_t|y_{<t}, x)$ ignores $y_{<t}$ and depends on x only via attention-based glimpses. We describe three experimental tricks we find helpful in §A.5.

4.2. Experimental results

Figure 1 shows the error rates for utterance-level g2p and p2g on the test set at various paired supervision levels (γ from 0.0005 to 1), both for a reference model trained only on the available paired data (blue bars) and for the proposed semi-supervised generative model (red bars). The proposed approach is able to make effective use of unpaired data to improve its predictions. Even very small amounts of paired data (5 minutes) are sufficient to effectively learn the association between spoken and written language. Numeric values are given in Table 6 and Table 7 in §D.4.

4.3. Ablations

Table 1 lists several ablation studies. These empirically investigate our hypothesis about time locality from §3 and the utility of the three training tricks described in §A.5. Making $p_\lambda(y|x)$ more powerful by allowing the $p_\lambda(y_t|y_{<t}, x)$ to depend directly on y_{t-1} or on a summary of $y_{<t}$ provided by the attention RNN state (see §D.3) degrades performance, especially at very low supervision levels, consistent with the discussion in §3. Pre-training the generative model was crucial to obtaining well-behaved training dynamics. Sampling at full temperature ($T = 1$) demonstrates the issue discussed in §A.5 with samples from KL-trained models and shows that lowering the temperature is an effective remedy. This issue is a particular problem at very low paired fractions, presumably due to the increased reliance on accurate $q_\nu(x|y)$ samples when learning from y-only data. Finally, the prediction metrics worsens when we update the prior weights using the ELBO as hypothesized in §A.5.

4.4. Samples

Here we present a sample of the model for utterance-level p2g task in Table 2. In this example `Buck Mulligan` is a rare proper noun that appears both in text-only and speech-only samples, but it is never observed in paired data. Therefore, the supervised-only model is unable to predict it correctly. However the proposed model is able to correctly associate between its phoneme and grapheme representations, although it is never presented with the pair.

5. RELATED WORK

The most closely related work is semi-supervised or unsupervised learning using distribution matching, where a divergence is minimized between the y-only data and y data obtained by mapping x-only data into the y domain using an implicit GAN-style $p_\lambda(y|x)$. Liu et al. (2018); Baevski et al. (2021); Liu et al. (2022a) use speech as x and phonemes derived from a text corpus as y to build unsupervised ASR models. It is interesting to note that Baevski et al. (2021) also impose a strict time locality assumption: successive phonemes output by the generator network are assumed to be conditionally independent given the speech waveform. We discussed the potential importance of this assumption in §3. A wide array of other related work is described in §E.

Ablation condition	supervised minutes (paired fraction γ)							
	5 min (γ=0.0005)		10 min (γ=0.001)		20 min (γ=0.002)		50 min (γ=0.005)	
	PER	CER	PER	CER	PER	CER	PER	CER
Only supervised	49.6	60.6	34.2	45.7	18.8	20.2	11.0	13.5
Proposed (semi-supervised)	**9.4**	**9.5**	**7.2**	**4.5**	5.6	4.2	**3.6**	**3.8**
y_t w/ receptive field to y_{t-1}	64.7	32.5	20.3	20.5	9.6	10.5	7.6	10.2
y_t w/ receptive field to $y_{<t}$	55.6	48.5	13.0	12.3	7.5	10.8	4.1	4.3
no $p_\lambda(x, y)$ pre-training	121.2	187.4	114.1	204.8	112.1	216.0	119.2	210.5
T = 0.8	73.2	217.0	70.5	5.6	**3.4**	**3.8**	**3.6**	4.1
T = 1	111.5	196.6	112.6	217.3	11.6	187.8	8.8	5.9
update $p_\lambda(x)$ with ELBO	100.6	173.8	63.3	18.3	9.7	7.0	3.9	4.4

Table 1. Ablation studies at various supervision rates. *Proposed* corresponds to $T = 0.5$, y_t w/o receptive field to $y_{<t}$, pre-trained $p_\lambda(x, y)$, and no update of $p_\lambda(x)$ with ELBO.

Sequence type	Sequence value
input phonemes	/sil **b V k m V l @ g @ n** sil w A: k @ N f O: r\ w @` d @ g E n sil r\ eI z d h I z h { n dz sil/
ground truth graphemes	***Buck Mulligan****, walking forward again, raised his hands.*
supervised-only prediction	***Buc-mullgaan****, walking forward again, raised his hands.*
semi-supervised prediction	***Buck Mulligan****, walking forward again, raised his hands.*

Table 2. Sample of an utterance-level p2g prediction.

6. DISCUSSION

We formulated semi-supervised seq2seq learning as a form of generative semi-supervised learning using approximate maximum likelihood estimation without any extra ad hoc losses. In this section we discuss several aspects of our approach and draw detailed connections.

Interestingly, the use of wake–sleep to train the variational posterior results in an approach with similarities to the *back-translation* method often used in unsupervised machine translation. We used l^{gen} and l^{var} from (1) and (2) as the training losses in our experiments, and as mentioned in §A.5 in practice we do not update $p_\lambda(x)$ when only y is observed. This means that $p_\lambda(x)$ only affects the training dynamics of $(p_\lambda(y|x), q_\nu(x|y))$ via the samples $x \sim p_\lambda(x)$ used to compute l^{var}, and so we could if we wished conceptually set $p_\lambda(x) = r(x)$ and sample ground truth x when training $q_\nu(x|y)$. We refer to this as the *prior trick*. In preliminary experiments we found this was sometimes very slightly less stable than our default approach but fundamentally still worked fine. If we use the prior trick and consider the case with no paired data ($\gamma = 0$) then, ignoring irrelevant additive constants, the training losses become

$$l^{\text{gen}} = \begin{cases} \log p_\lambda(y|x^{\text{var}}) & \text{if } y \text{ obs} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$l^{\text{var}} = \begin{cases} \log q_\nu(x|y^{\text{gen}}) & \text{if } x \text{ obs} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $x^{\text{var}} \sim q_\nu(x|y)$ and $y^{\text{gen}} \sim p_\lambda(y|x)$. This is extremely similar to iterative back-translation. There remain a few important differences. Firstly, we sample x^{var} and y^{gen}, albeit at reduced temperature (§A.5), rather than using a form of most probable decoding such as beam search. Sampling ensures probabilistic consistency and may help with calibration of the model's predictions, though we do not evaluate that here. Tjandra et al. (2019) explore several decoding strategies including greedy and beam search for back-translation. Secondly, while (4) and (5) are symmetric with respect to the role of $p_\lambda(y|x)$ and $q_\nu(x|y)$, our model architecture choices are not: we use a time-local $p_\lambda(y|x)$ and flexible $q_\nu(x|y)$. Making both flexible can lead to problems with identifiability. Making both time-local can lead to pathologies in the training dynamics. As a slightly simplistic example, if $p_\lambda(y|x) = \prod_t p_\lambda(y_t|x_t)$ and $q_\nu(x|y) = \prod_t q_\nu(x_t|y_t)$ then it can be shown then training ignores correlations over time in x and y even if the true $p_\lambda(y|x)$ and $q_\nu(x|y)$ are of this form. It would be interesting to experiment with tweaks to back-translation inspired by the above derivation of it as a specific instance of a probabilistically principled approach.

Our approach resembles the noisy channel decipherment model by Ravi & Knight (2011) for unsupervised translation, which interestingly also makes a very strong time local independence assumption of word-to-word substitution in the channel. Klejch et al. (2021) has adapted their method for zero-resource cross-lingual ASR, while maintaining similar conditional independence assumption, permitting token insertion and deletion via finite-state transducers. Our approach here is a modern take on the decipherment idea, but with flexible neural models for prior and decoder.

Accurate density modeling is crucial for this type of generative modeling. For instance uncalibrated $\log p_\lambda(y|x)$ can adversely affect the balance in y-only term of l^{gen}. To avoid having to worry about the challenges of modeling high-dimensional continuous quantities accurately, we have only experimented with phonemes and graphemes using categorical output distributions and evaluated utterance-level g2p/p2g tasks. These are not common tasks with established state-of-the-art performance, especially in semi-supervised setups. The closest comparison is the fully-supervised utterance-level g2p prediction accuracy reported by Juzová et al. (2019), which is on par with our results for $\lambda = 1.0$. These experimental tasks served as nice test bed for this novel approach. A natural question is how these results hold when we move to end-to-end ASR and TTS, for which we would need accurate speech density estimation. Alternatively, given the success of tokenized speech representation a natural extension of this work is to use such representations (Lakhotia et al., 2021; Zeghidour et al., 2021; Défossez et al., 2022). Also, the masked language model (MLM)-based pre-trained models under uniform masking regime (Ghazvininejad et al., 2019) can be interpreted as probability distributions estimated with maximum likelihood, and hence it can be directly plugged into our formulation as $p_\lambda(x)$, making this generative modeling approach also nicely amenable to the commonly adopted pre-training / fine-tuning workflow.

Similar conditional independence structure appears in other tasks (e.g., machine translation) and the approach and discussions here could be more widely applicable.

References

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pp. 173–182. PMLR, 2016.

Ao, J., Wang, R., Zhou, L., Liu, S., Ren, S., Wu, Y., Ko, T., Li, Q., Zhang, Y., Wei, Z., et al. SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing. *arXiv preprint arXiv:2110.07205*, 2021.

Baevski, A., Schneider, S., and Auli, M. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*, 2020a.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020b.

Baevski, A., Hsu, W.-N., Conneau, A., and Auli, M. Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, 34, 2021.

Baker, J. Stochastic modeling for automatic speech recognition. *Speech Recognition*, pp. 521–542, 1975.

Bapna, A., Chung, Y.-a., Wu, N., Gulati, A., Jia, Y., Clark, J. H., Johnson, M., Riesa, J., Conneau, A., and Zhang, Y. SLAM: A unified encoder for speech and language modeling via speech-text joint pre-training. *arXiv preprint arXiv:2110.10329*, 2021.

Battenberg, E., Skerry-Ryan, R., Mariooryad, S., Stanton, D., Kao, D., Shannon, M., and Bagby, T. Location-relative attention mechanisms for robust long-form speech synthesis. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6194–6198. IEEE, 2020.

Beal, M. J. and Ghahramani, Z. The variational Kalman smoother. *Gatsby Computational Neuroscience Unit, University College London, Tech. Rep. GCNU TR01-003*, 2000.

Bishop, C. M. *Pattern recognition and machine learning*. Springer, 2006.

Borsos, Z., Marinier, R., Vincent, D., Kharitonov, E., Pietquin, O., Sharifi, M., Teboul, O., Grangier, D., Tagliasacchi, M., and Zeghidour, N. AudioLM: a language modeling approach to audio generation. *arXiv preprint arXiv:2209.03143*, 2022.

Chan, W., Jaitly, N., Le, Q., and Vinyals, O. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4960–4964. IEEE, 2016.

Chan, W., Park, D., Lee, C., Zhang, Y., Le, Q., and Norouzi, M. SpeechStew: Simply mix all available speech recognition data to train one large neural network. *arXiv preprint arXiv:2104.02133*, 2021.

Chen, Z., Rosenberg, A., Zhang, Y., Zen, H., Ghodsi, M., Huang, Y., Emond, J., Wang, G., Ramabhadran, B., and Mengibar, P. J. M. Semi-supervision in ASR: Sequential mixmatch and factorized TTS-based augmentation. In *Proc. Interspeech*, pp. 736–740, 2021a.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Wang, G., and Moreno, P. Injecting text in self-supervised speech pretraining. *arXiv preprint arXiv:2108.12226*, 2021b.

Chen, Z., Chen, S., Wu, Y., Qian, Y., Wang, C., Liu, S., Qian, Y., and Zeng, M. Large-scale self-supervised speech representation learning for automatic speaker verification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6147–6151. IEEE, 2022a.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Moreno, P., Bapna, A., and Zen, H. MAESTRO: Matched speech text representations through modality matching. *arXiv preprint arXiv:2204.03409*, 2022b.

Chung, Y.-A., Wang, Y., Hsu, W.-N., Zhang, Y., and Skerry-Ryan, R. Semi-supervised training for improving data efficiency in end-to-end speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6940–6944. IEEE, 2019.

Chung, Y.-A., Zhang, Y., Han, W., Chiu, C.-C., Qin, J., Pang, R., and Wu, Y. W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. *arXiv preprint arXiv:2108.06209*, 2021.

Cooper, D. B. and Freeman, J. H. On the asymptotic improvement in the out-come of supervised learning provided by additional nonsupervised learning. *IEEE Transactions on Computers*, 100(11):1055–1063, 1970.

Défossez, A., Copet, J., Synnaeve, G., and Adi, Y. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

Denes, P. B., Denes, P., and Pinson, E. *The speech chain*. Macmillan, 1993.

Ghazvininejad, M., Levy, O., Liu, Y., and Zettlemoyer, L. Mask-predict: Parallel decoding of conditional masked language models. In *Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., et al. Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech*, 2020.

Han, W., Zhang, Z., Zhang, Y., Yu, J., Chiu, C.-C., Qin, J., Gulati, A., Pang, R., and Wu, Y. Contextnet: Improving convolutional neural networks for automatic speech recognition with global context. In *Interspeech*, 2020.

Hayashi, T., Watanabe, S., Zhang, Y., Toda, T., Hori, T., Astudillo, R., and Takeda, K. Back-translation-style data augmentation for end-to-end ASR. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 426–433. IEEE, 2018.

Hayashi, T., Watanabe, S., Toda, T., Takeda, K., Toshniwal, S., and Livescu, K. Pre-trained text embeddings for enhanced text-to-speech synthesis. In *INTERSPEECH*, pp. 4430–4434, 2019.

Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Hori, T., Astudillo, R., Hayashi, T., Zhang, Y., Watanabe, S., and Le Roux, J. Cycle-consistency training for end-to-end speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6271–6275. IEEE, 2019.

Hsu, W.-N., Tsai, Y.-H. H., Bolte, B., Salakhutdinov, R., and Mohamed, A. HuBERT: How much can a bad teacher benefit ASR pre-training? In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6533–6537. IEEE, 2021.

Jelinek, F. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976.

Jia, Y., Zen, H., Shen, J., Zhang, Y., and Wu, Y. Png BERT: augmented bert on phonemes and graphemes for neural TTS. *arXiv preprint arXiv:2103.15060*, 2021.

Juzová, M., Tihelka, D., and Vít, J. Unified language-independent DNN-based G2P converter. In *Interspeech*, pp. 2085–2089, 2019.

Kernighan, M. D., Church, K. W., and Gale, W. A. A spelling correction program based on a noisy channel model. In *COLING 1990*, 1990.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Kingma, D. P., Mohamed, S., Jimenez Rezende, D., and Welling, M. Semi-supervised learning with deep generative models. *Advances in Neural Information Processing Systems*, 27, 2014.

Klejch, O., Wallington, E., and Bell, P. Deciphering speech: a zero-resource approach to cross-lingual transfer in ASR. *arXiv preprint arXiv:2111.06799*, 2021.

Lakhotia, K., Kharitonov, E., Hsu, W.-N., Adi, Y., Polyak, A., Bolte, B., Nguyen, T.-A., Copet, J., Baevski, A., Mohamed, A., and Dupoux, E. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354, 2021.

Li, B., Sainath, T. N., Pang, R., and Wu, Z. Semi-supervised training for end-to-end models via weak distillation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2837–2841. IEEE, 2019.

Liu, A. H., Hsu, W.-N., Auli, M., and Baevski, A. Towards end-to-end unsupervised speech recognition. *arXiv preprint arXiv:2204.02492*, 2022a.

Liu, A. H., Lai, C.-I. J., Hsu, W.-N., Auli, M., Baevskiv, A., and Glass, J. Simple and effective unsupervised speech synthesis. *arXiv preprint arXiv:2204.02524*, 2022b.

Liu, D.-R., Chen, K.-Y., Lee, H.-y., and Lee, L.-s. Completely unsupervised phoneme recognition by adversarially learning mapping relationships from audio embeddings. *arXiv preprint arXiv:1804.00316*, 2018.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. LibriSpeech: an ASR corpus based on public domain audio books. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., and Tran, D. Image transformer. In *International Conference on Machine Learning*, pp. 4055–4064. PMLR, 2018.

Rabiner, L. R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Ravi, S. and Knight, K. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 12–21, 2011.

Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. FastSpeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

Rosenberg, A., Zhang, Y., Ramabhadran, B., Jia, Y., Moreno, P., Wu, Y., and Wu, Z. Speech recognition with augmented synthesized speech. In *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*, pp. 996–1002. IEEE, 2019.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. wav2vec: Unsupervised pre-training for speech recognition. In *Interspeech*, 2019.

Sennrich, R., Haddow, B., and Birch, A. Improving neural machine translation models with monolingual data. *Association for Computational Linguistics*, 2016.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4779–4783. IEEE, 2018.

Shor, J., Jansen, A., Han, W., Park, D., and Zhang, Y. Universal paralinguistic speech representations using self-supervised conformers. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3169–3173. IEEE, 2022.

Tjandra, A., Sakti, S., and Nakamura, S. Listening while speaking: Speech chain by deep learning. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 301–308. IEEE, 2017.

Tjandra, A., Sakti, S., and Nakamura, S. End-to-end feedback loss in speech chain framework via straight-through estimator. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6281–6285. IEEE, 2019.

Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., et al. Tacotron: Towards end-to-end speech synthesis. In *Interspeech*, 2017.

Weiss, R. J., Skerry-Ryan, R., Battenberg, E., Mariooryad, S., and Kingma, D. P. Wave-tacotron: Spectrogram-free end-to-end text-to-speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5679–5683. IEEE, 2021.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J., Jia, Y., Chen, Z., and Wu, Y. LibriTTS: A corpus derived from LibriSpeech for text-to-speech. In *Interspeech*, 2019.

A. MODEL: DETAILS

In this section we give more details of our broad approach to semi-supervised learning and how we perform learning and inference. One of the advantages of generative modeling is that it provides a principled way to use unpaired data (Cooper & Freeman, 1970).

A.1. Semi-supervised generative modeling

Our approach to semi-supervised learning is straightforward: it is simply maximum likelihood estimation in the presence of missing data. We formulate an overall generative model which represents both how the pair (x, y) of sequences is generated and how data becomes missing, and then maximize the likelihood of the training partition of the dataset under this model. While this principled approach is relatively well-known, we take time in this section to establish terminology and notation and to review the conceptual framing in detail.

We first describe conventional supervised learning of the joint distribution of two sequences. In this case we have a dataset where each element is a pair (x, y) of a sequence $x = [x_s]_{s=0}^{S-1}$ and a sequence $y = [y_t]_{t=0}^{T-1}$ of possibly different length. For example, for applications such as speech synthesis or speech recognition involving the relationship between text and speech, x might be a sequence of graphemes and y a sequence of mel spectrogram frames derived from corresponding speech audio. We assume this dataset was generated by sampling i.i.d. from a *true distribution* $r(x, y)$. We aim to learn the parameters λ of a parametric probabilistic model $p_\lambda(x, y)$ to minimize

$$l_\lambda = -\mathbb{E}_r[\log p_\lambda(x, y)] = -\sum_{x,y} r(x, y) \log p_\lambda(x, y) \tag{6}$$

In practice we replace the expectation over all possible data with stochastic minibatches drawn from the training partition of the dataset, yielding a form of maximum likelihood estimation.

We formalize the semi-supervised and unsupervised cases as follows. We assume there is an underlying true distribution $r(m, x, y)$ where x and y are two sequences as before and the *missingness* $m \in \{0, 1, 2\}$ is an indicator variable determining what is observed. The observed dataset element u is then $(0, x)$, $(1, y)$ or $(2, x, y)$ depending on whether $m = 0$ (only x observed), $m = 1$ (only y observed) or $m = 2$ (both x and y observed). To extend the probabilistic model to include m, we assume the data distribution does not depend on m, that is $p_\lambda(m, x, y) = p_\lambda(m)p_\lambda(x, y)$, even though this may only be approximately true in practice. For example, we assume that the distribution of text in the text-only data is the same as the distribution of text in the paired data. Thus $p_\lambda(u = (0, x)) = p_\lambda(m = 0)p_\lambda(x)$, $p_\lambda(u = (1, y)) = p_\lambda(m = 1)p_\lambda(y)$ and $p_\lambda(u = (2, x, y)) = p_\lambda(m = 2)p_\lambda(x, y)$. We now aim to learn the parameters λ to minimize

$$\begin{aligned} l_\lambda &= -\mathbb{E}_r[\log p_\lambda(u)] = -\sum_u r(u) \log p_\lambda(u) &(7)\\ &= -\mathbb{E}_r\left[\log p_\lambda(m) + \delta_{m0} \log p_\lambda(x) + \delta_{m1} \log p_\lambda(y) + \delta_{m2} \log p_\lambda(x, y)\right] &(8)\end{aligned}$$

We again replace the expectation over all possible data with stochastic minibatches drawn from the training partition of the dataset in practice. Our formulation is thus simply maximum likelihood estimation in the presence of missing data.

A.2. Noisy channel sequential model

We assume a model of the form $p_\lambda(x, y) = p_\lambda(x)p_\lambda(y|x)$. This is sometimes known as a *noisy channel model*, viewing $p_\lambda(y|x)$ as a channel which does not preserve information in x perfectly as it is transformed stochastically to y. In preliminary experiments models of this form performed better than assuming x and y are each generated from a shared sequential latent variable z, that is $p_\lambda(x, y) = \sum_z p_\lambda(z)p_\lambda(x|z)p_\lambda(y|z)$. We use autoregressive models for $p_\lambda(x)$ and $p_\lambda(y|x)$, so roughly speaking

$$p_\lambda(x) = \prod_s p_\lambda(x_s|x_{0:s-1}) \tag{9}$$

$$p_\lambda(y|x) = \prod_t p_\lambda(y_t|y_{0:t-1}, x) \tag{10}$$

However we must also model the length S and T of the sequences $x = [x_s]_{s=0}^{S-1}$ and $y = [y_t]_{t=0}^{T-1}$, and we do this using a sequence of binary end-of-sequence decisions. Taking the case of $p_\lambda(x)$, let $e = [e_s]_{s=0}^{S}$ where $e_s = \mathbb{1}\{S \leq s\}$. For example, if $x = [\mathtt{a}, \mathtt{b}, \mathtt{c}]$ then $e = [0, 0, 0, 1]$. We set

$$p_\lambda(x) = \left(\prod_{s=0}^{S-1} p_\lambda(e_s = 0 | e_{0:s-1} = 0, x_{0:s-1}) p_\lambda(x_s | e_{0:s} = 0, x_{0:s-1}) \right) p_\lambda(e_S = 1 | e_{0:S-1} = 0, x_{0:S-1}) \tag{11}$$

In the case where x is discrete, using a sequence of binary end-of-sequence decisions is essentially equivalent to appending a special end-of-sequence token to x, but the former has the advantage of also being applicable when x is continuous.

A.3. Approximate marginalization using variational inference

We now discuss how to perform training and inference. We approximate the marginals required to do this exactly using variational inference.

For training using (8), we require the marginals $p_\lambda(x) = \sum_y p_\lambda(x, y)$ and $p_\lambda(y) = \sum_x p_\lambda(x, y)$. In models of the form $p_\lambda(x, y) = p_\lambda(x) p_\lambda(y|x)$, we obtain $p_\lambda(x)$ for free. However we still need to compute $p_\lambda(y)$. This can be done analytically in simple models such as hidden Markov models and finite state transducers, but in our main experiments $p_\lambda(x)$ is an arbitrarily powerful neural language model, making analytic marginalization intractable. We instead use a variational lower bound on $\log p_\lambda(y)$.

The variational lower bound we use is just a variant of the conventional ELBO (Beal & Ghahramani, 2000). This can be derived in quite a concise and informative way using KL divergences. Define a joint distribution $q_\nu(x, y) = r(y) q_\nu(x|y)$. Thus $q_\nu(y) = r(y)$. Then we have

$$\begin{aligned} \mathrm{KL}[q_\nu(x, y) \,\|\, p_\lambda(x, y)] &= \mathrm{KL}[q_\nu(y) \,\|\, p_\lambda(y)] + \mathrm{KL}[q_\nu(x, y) \,\|\, q_\nu(y) p_\lambda(x|y)] & (12) \\ &= \mathrm{KL}[r(y) \,\|\, p_\lambda(y)] + \mathrm{KL}[r(y) q_\nu(x|y) \,\|\, r(y) p_\lambda(x|y)] & (13) \end{aligned}$$

By the non-negativity of the KL divergence, we have

$$\mathrm{KL}[r(y) q_\nu(x|y) \,\|\, p_\lambda(x, y)] \geq \mathrm{KL}[r(y) \,\|\, p_\lambda(y)] \tag{14}$$

with equality if and only if $q_\nu(x|y) = p_\lambda(x|y)$ for all x and y.[†] Negative and adding the entropy $-\mathbb{E}_r[\log r(y)]$ to both sides yields

$$\sum_{x,y} r(y) q_\nu(x|y) \big(\log p_\lambda(x, y) - \log q_\nu(x|y) \big) \leq \sum_y r(y) \log p_\lambda(y) \tag{15}$$

the ELBO variant To obtain a tractable objective function amenable to gradient-based optimization, we may therefore replace $\log p_\lambda(y)$ in (8) with the expected value of $\log p_\lambda(x, y) - \log q_\nu(x|y)$ under $r(y) q_\nu(x|y)$, giving (1).

A.4. Wake–sleep algorithm

In many cases of interest the sequence x has discrete values. In this case some approach other than simple reparameterized sampling (Kingma & Welling, 2014) is required to compute the gradient of $\mathrm{KL}[r(y) q_\nu(x|y) \,\|\, p_\lambda(x, y)]$ with respect to ν. REINFORCE (Williams, 1992), the wake–sleep algorithm (Hinton et al., 1995), RELAX (Grathwohl et al., 2018) and many other approaches have been proposed for discrete x and could be adapted for our use case involving sequential variational posteriors $q_\nu(x|y)$. In this section we describe a variant of the wake–sleep algorithm used in our experiments.

We modify the loss used to train the variational posterior. Instead of minimizing $\mathrm{KL}[r(y) q_\nu(x|y) \,\|\, p_\lambda(x, y)]$ with respect to the encoder parameters ν, we minimize $\mathrm{KL}[p_\lambda(x, y) \,\|\, r(y) q_\nu(x|y)]$ with respect to ν. This involves sampling (x, y) pairs from the *generative* model $p_\lambda(x, y)$ and training $q_\nu(x|y)$ with maximum likelihood estimation on the generated data. Ignoring irrelevant constants, minimizing $\mathrm{KL}[p_\lambda(x, y) \,\|\, r(y) q_\nu(x|y)]$ amounts to maximizing

$$\sum_{x,y} p_\lambda(x, y) \log q_\nu(x|y) \tag{16}$$

as in (2). We continue to train the generative model parameters λ as before. The λ updates and ν updates performed during gradient-based optimization correspond to the wake phase and sleep phase respectively (Hinton et al., 1995). The

[†]Technically for all x and y with $r(y) q_\nu(x|y) > 0$, but it is reasonable to assume that no sequence y is truly impossible under the true distribution $r(y)$ and that we choose a variational posterior $q_\nu(x|y)$ which never assigns a probability of exactly zero to any x.

conventional ELBO and the sleep-phase loss have the same non-parametric optimal variational posterior $\hat{q}(x|y) = p_\lambda(x|y)$, so the two methods would produce the same result if a sufficiently flexible and easy to optimize model was trained to convergence. The two approaches place different demands on q_ν and p_λ: the conventional approach requires samples from $q_\nu(x|y)$ to be reparameterized while the sleep-phase loss requires samples from $p_\lambda(x, y)$.

The use of different objectives for different parts of the model is reminiscent of GAN training, but note that here the losses are cooperative rather than adversarial, in the sense that making the variational posterior optimal improves both the variational loss and the generative loss, whereas making the critic optimal in classic GAN training makes the generator loss worse. Nevertheless, there is no guarantee that the training dynamics of the (generative, variational) system are convergent in general.[‡]

A.5. Training tricks

We find three experimental tricks helpful for training. Firstly, samples from autoregressive models can suffer from small errors compounding over time, particularly when trained with maximum likelihood estimation / KL. This only weakly penalizes unrealistic next-step samples because KL is a "covering" rather than "mode-seeking" divergence (Bishop, 2006, Section 10.1.2). A common trick for both non-autoregressive (Parmar et al., 2018; Kingma & Dhariwal, 2018) and autoregressive (Weiss et al., 2021) models is to adjust the temperature of the distribution. The prior, decoder and variational posterior are all trained with KL, and we apply temperature adjustment when sampling from these models during both training and decoding. For example, for the variational posterior we recursively sample from $\frac{1}{Z_\nu(x_{0:t-1}, y)}(q_\nu(x_t|x_{0:t-1}, y))^{\frac{1}{T}}$ instead of $q_\nu(x_t|x_{0:t-1}, y)$, where Z_ν is the partition function ensuring a normalized distribution. Typically $T = 0.5$. Secondly, at random initialization the generative model and variational posterior are both very suboptimal, and the noisy gradients from the β term of l^{gen} may swamp the small but consistent signal from the paired data γ term when training the decoder. To alleviate this, we pre-train with the β term omitted from l^{gen}, effectively ignoring the y-only data. Finally, we optionally ignore the ELBO term throughout training when updating the prior $p_\lambda(x)$. In the regime where α is small this could prevent the model learning important information about $r(x)$ present in the y-only data, but in the regime we consider here where there is plenty of x-only data, it slightly helps to stabilize training.

B. IDENTIFIABILITY

Learning with no paired data is particularly challenging. In this section we discuss when we might expect this to be possible, focusing on *time locality* as a guiding principle.

Definition B.1 (Identifiability given no paired data)**.** We say a parametric family $\{p_\lambda : \lambda \in \Lambda\}$ of joint distributions over (x, y) is *identifiable given no paired data* if matching the marginals implies matching the joint, that is if $p_\lambda(x) = p_{\lambda_\text{T}}(x)$ for all x and $p_\lambda(y) = p_{\lambda_\text{T}}(y)$ for all y implies $p_\lambda(x, y) = p_{\lambda_\text{T}}(x, y)$ for all x and y ($\lambda, \lambda_\text{T} \in \Lambda$).

The above definition formalizes what it means to learn perfectly from plentiful unpaired-only data. We may think of λ as the parameters being learned and λ_T as the true parameters. Clearly we must restrict the joint distribution $p_\lambda(x, y)$ to have any hope of identifiability given no paired data: if $p_\lambda(y|x)$ can capture any distribution over y without using x (e.g. having a receptive field to past steps $y_{<t}$ in Table 1) then it may attain a perfect marginal $p_\lambda(y)$ without learning anything about the true relationship between x and y.

We now present a complete analysis of identifiability given no paired data in a simplified version of our full setup, showing that time locality is generically sufficient to ensure identifiability. This helps motivate time locality and guides our intuition around identifiability in more complicated cases.

We consider a Markovian prior and time-independent and time-synchronous decoder with no paired data available. In this case $p_\lambda(x, y) = \prod_t p_\lambda(x_t|x_{t-1}) p_\lambda(y_t|x_t)$. Let $B_{ij} = p_\lambda(x_0 = i, x_1 = j)$, $O_{ip} = p_\lambda(y_t = p|x_t = i)$, $D_{pq} = p_\lambda(y_0 = p, y_1 = q)$ and $\mathbb{1}$ be a vector of ones. We assume that the prior is a stationary distribution, that is $B\mathbb{1} = B^\mathsf{T}\mathbb{1} = b$ and that $b_i > 0$ for all i. The prior is easy to learn from unpaired data, and so we assume $r(x_0 = i, x_1 = j) = B_{ij}$. Let $C_{pq} = r(y_0 = p, y_1 = q)$ and $c = C\mathbb{1} = C^\mathsf{T}\mathbb{1}$. In this case we can conveniently express the relationship between the y marginals and x marginals as a matrix multiplication $D = O^\mathsf{T} B O$.

[‡]If the learning rate used for the generative parameters λ is set sufficiently small relative to the learning rate used for the variational parameters ν, and the variational posterior is sufficiently flexible, then the variational posterior is able to remain essentially optimal throughout training and so the training dynamics are effectively just gradient descent on (1) with respect to λ, which has well-behaved training dynamics.

We first consider the case where O is a permutation matrix, corresponding to a *substitution cipher*. We assume x is English text represented as a series of graphemes. For example, the ciphertext y might be `wi jtvwjpvwjbhjwi jgvw`, corresponding to English plaintext x, here `the cat sat on the mat`. It is well-known that this simple cipher can be broken by frequency analysis, by tabulating the frequency of grapheme n-grams in the ciphertext and looking for grapheme n-grams with similar frequencies in conventional English text. We may codify this by considering the singular value decompositions of B and C. We now show that as long as the singular values of B are distinct and non-zero then we can completely recover O and have identifiability given no paired data. Compute the singular value decomposition of the plaintext bigram frequencies B and the ciphertext bigram frequencies C as

$$B = U_x \Lambda_x {V_x}^\intercal \tag{17}$$

$$C = U_y \Lambda_y {V_y}^\intercal \tag{18}$$

where U_x, V_x, U_y, V_y are real orthogonal matrices and Λ_x and Λ_y are real diagonal matrices with entries increasing along the diagonal. We know that C must also equal $O^\intercal B O = (O^\intercal U_x)\Lambda_x(O^\intercal V_x)^\intercal$ which is also a singular value decomposition of C. Standard results on the uniqueness of the singular value decomposition, which can be obtained by considering the eigenvalues and eigenvectors of $CC^\intercal$ and $C^\intercal C$, show that $\Lambda_x = \Lambda_y = \Lambda$ and that, if the singular values are all distinct and non-zero, then the left and right singular vectors are determined up to sign, that is $U_y = O^\intercal U_x S$ and $V_y = O^\intercal V_x T$ for two diagonal matrices S and T with 1s or -1s along their diagonal. Thus $O = U_x S {U_y}^\intercal = V_x T {V_y}^\intercal$. Since $O\mathbb{1} = \mathbb{1}$, we have $S{U_y}^\intercal \mathbb{1} = {U_x}^\intercal \mathbb{1}$, which allows us to recover S, and similarly T. Thus O is identifiable given no paired data in this case as long as the singular values are distinct.

Secondly we consider the case where O is not restricted to be a permutation matrix but where the x and y alphabets both have size two, say $x_s, y_t \in \{0, 1\}$. Since $O\mathbb{1} = \mathbb{1}$, there are only two degrees of freedom in O, say

$$O = \begin{bmatrix} \eta & 1-\eta \\ \zeta & 1-\zeta \end{bmatrix} \tag{19}$$

We first consider cases where we do not have identifiability, which may be particularly helpful for building general intuition. The first degenerate case is where B is low rank, that is $B = bb^\intercal$ and x_0 and x_1 are independent. In this case O is never identifiable, since any (η, ζ) on the line $\eta b_0 + \zeta b_1 = c_0$ results in the same unigram marginal $p_\lambda(y_0)$ and so the same overall marginal $p_\lambda(y)$ due to independence over time. This is a simple example of needing correlations over time in x that are longer than the decoder can model on its own in order to have identifiability. The second degenerate case we consider is where $b_0 = \frac{1}{2}$. In this case

$$B = \begin{bmatrix} B_{00} & B_{01} \\ B_{01} & B_{00} \end{bmatrix} \tag{20}$$

This obeys the symmetry that swapping 0s and 1s does not change the probability of a sequence under the prior. Intuitively this means we have no way to distinguish which x symbol maps to a given y symbol, just like in the case where x is a latent variable which is never observed. Formally (η, ζ) and (ζ, η) result in the same marginal $p_\lambda(y)$ for all y. Technically we do still have identifiability if $\eta = \zeta$, but this case is practically uninteresting because it means x and y are completely independent. Otherwise we do not have identifiability when $b_0 = \frac{1}{2}$. By considering sequences of length two and three, we now show that if B is full rank and $b_0 \neq \frac{1}{2}$ then we do have identifiability given no paired data. We first consider sequences of length two. We suppose that $O^\intercal B O = {O_\mathrm{T}}^\intercal B O_\mathrm{T}$ for some O, O_T and ask whether this implies $O = O_\mathrm{T}$. The expression $O^\intercal B O$ is a quadratic function of η and ζ. By explicitly expanding in terms of η and ζ, it may be verified that the only possible solutions are

$$O = \begin{bmatrix} c_0 & c_1 \\ c_0 & c_1 \end{bmatrix} \pm \det(O_\mathrm{T}) \begin{bmatrix} b_1 & -b_1 \\ -b_0 & b_0 \end{bmatrix} \tag{21}$$

where the determinant $\det(O_\mathrm{T})$ is also equal to $\det(C)/\det(B)$. In some cases one of these solutions will have a negative entry and so not represent a valid decoder, in which case we have identifiability. This is more likely when O_T has large determinant. As before we also technically have identifiability in the uninteresting case where $\eta = \zeta$. Thus in general based on sequences of length two alone we cannot uniquely identify O. Now consider sequences of length three. Define the trigram distributions

$$B_{ijk} = p_\lambda(x_0 = i, x_1 = j, x_2 = k) \tag{22}$$
$$C_{pqr} = r(y_0 = p, y_1 = q, y_2 = r) \tag{23}$$
$$D_{pqr} = p_\lambda(y_0 = p, y_1 = q, y_2 = r) = \sum_{i,j,k} B_{ijk} O_{ip} O_{jq} O_{kr} \tag{24}$$

We suppose that $\sum_{i,j,k} B_{ijk} O_{ip} O_{jq} O_{kr} = \sum_{i,j,k} B_{ijk} (O_\mathrm{T})_{ip} (O_\mathrm{T})_{jq} (O_\mathrm{T})_{kr}$ for some O, O_T and ask whether this implies $O = O_\mathrm{T}$. We only need to worry about distinguishing the two solutions in (21). By plugging in these two solutions into (24) it can be shown that if

$$\sum_{i,j,k} B_{ijk} (b^\perp)_i (b^\perp)_j (b^\perp)_k \neq 0 \tag{25}$$

where $b^\perp = [-b_1, b_0]$ then $O = O_\mathrm{T}$. The above holds for general prior distributions $p_\lambda(x)$. If $p_\lambda(x)$ is Markovian then $B_{ijk} = B_{ij} B_{jk} / b_j$ and the condition (25) is equivalent to $b_0 \neq \frac{1}{2}$. Thus if B is full rank and $b_0 \neq \frac{1}{2}$ then we have identifiability given no paired data. This holds for any choice of O specifying the decoder. The general pattern in this simple case is that the time locality assumption is sufficient to ensure identifiability unless the marginal $p_\lambda(x)$ obeys one of a finite list of a specific symmetries that make identification impossible.

C. ADDITIONAL EXPERIMENT: BREAKING A SUBSTITUTION CIPHER

In this section, we apply the proposed approach to the task of breaking a substitution cipher as outlined in §B. In this case the marginal $p_\lambda(y)$ is fully tractable and we do not have to use a variational approximation. This investigates the ability of the proposed approach to learn from no paired data.

C.1. Experimental setup

The experimental setup is as follows. Our training data is a randomly chosen subset of 2 000 utterances from the LibriTTS corpus (Zen et al., 2019). We derive plaintext x by lowercasing and removing punctuation from the text transcript and pass this through a fixed permutation of the 27-character grapheme alphabet to obtain ciphertext y. This yields a total of roughly 140 000 grapheme tokens. We set $\gamma = 0$ and $\alpha = \beta = \frac{1}{2}$, using half the data for collecting bigram frequencies B_{ij} on the plaintext and half for collecting bigram frequencies C_{pq} on the ciphertext. Length 2 statistics appear to be sufficient for this task. We learn the observation matrix $O \in \mathbb{R}^{27 \times 27}$, parameterized in terms of its logits using a softmax for each row to ensure that O is stochastic, to minimize the bigram loss $l = \sum_{p,q} C_{pq} \log([O^\intercal B O]_{pq})$. This is the cross-entropy for the marginal distribution of y. To initialize O, we take the *unigram initialization* $O = \mathbb{1} c^\intercal$, where c is the observed ciphertext unigram frequencies, and apply a small random normal perturbation to the logits to break symmetry since the unigram initialization is a stationary point. For optimization we use an initial phase of stochastic gradient descent with learning rate 10 to get in roughly the right region of parameter space followed by Adam (Kingma & Ba, 2014) with learning rate 0.01 to converge to the precise local optimum, since we find this works substantially better than either optimizer on its own.

C.2. Experimental results

Training finds the correct mapping O between plaintext and ciphertext for roughly 80% of training runs. Training success is very binary, typically either succeeding essentially perfectly with near-zero $\mathrm{KL}[r(y) \,\|\, p_\lambda(y)]$ and decoding error rate, or failing with a large KL divergence value and error rate. This suggests that the training loss can potentially serve as an indicator of when training fails. These results demonstrate that learning from zero paired data is possible with the proposed approach, as well as highlighting the substantial challenge that local optima present in this regime. Klejch et al. (2021) has reported similar initialization issues for a decipherment model. To investigate the effect of random initialization, we perform 50 training runs, sort by the achieved training loss, and plot the training loss $\mathrm{KL}[r(y_0, y_1) \,\|\, p_\lambda(y_0, y_1)]$ and decoding error rate in Figure 2. The binary nature of success or failure mentioned in §C is clearly visible. This is somewhat encouraging since it suggests that the training loss can potentially serve as a reliable detector of when to discard a training run, at least when the generative model matches the true generating process well as here.

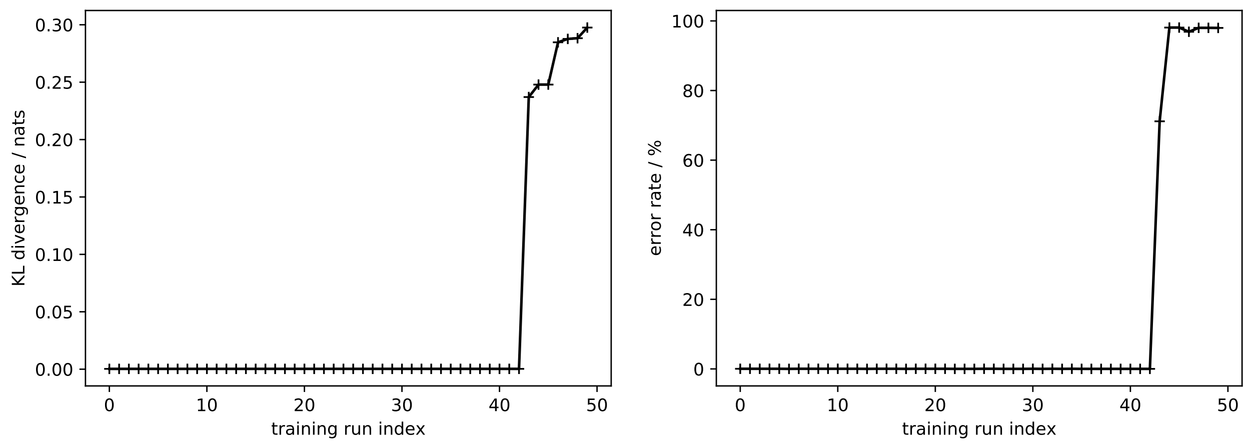

Figure 2. The performance of the learned models across 50 random training runs sorted by increasing final training loss. Performance is measured by the training loss, which is the bigram KL divergence $\mathrm{KL}[r(y_0, y_1) \,\|\, p_\lambda(y_0, y_1)]$, and the decoding error rate. Training success is very binary and there is a strong correlation between training loss and error rate.

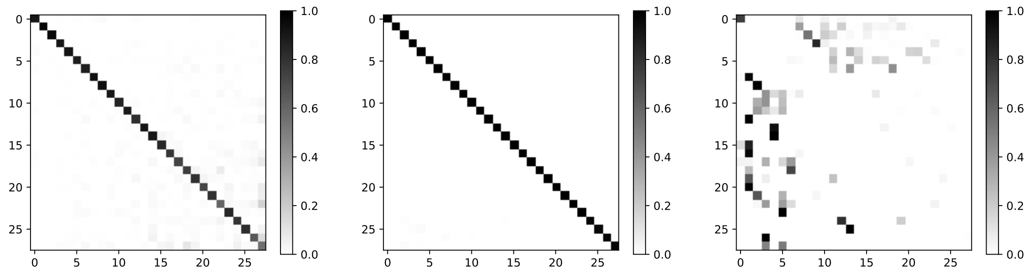

Figure 3. Examples of learned O for singular value decomposition (left), a successful gradient descent run (middle) and a failed gradient descent run (right). The vertical axis is the plaintext grapheme x_t and the horizontal axis is the ciphertext grapheme y_t (with columns reordered so that the true mapping is the identity). Thus each row represents the learned probability distribution $p_\lambda(y_t|x_t)$ for a particular x_t. The SVD solution is not perfect due to its sensitivity to the plaintext and ciphertext coming from distinct utterances. The failed run finds a suboptimal local minimum in the training loss, and essentially the same O is found by multiple different failed training runs. In this case the failed run appears to have learned to map vowels to consonants and consonants to vowels.

Examples of the O learned when training succeeds and fails are shown in Figure 3. The symbol table used is

```
0000000000111111111122222222
0123456789012345678901234567
^␣eaoiutnhsrdlmcwfygpbvkxqjz
```

where ^ is a start-of-sequence and end-of-sequence symbol and ␣ is the space character. Note the grouping of vowels then consonants. The matrix O learned at various stages of training for a successful training run is shown in Figure 4. For coherence in this case Adam was used throughout. This shows how the association for common symbols is learned earlier in training, potentially unlocking additional known contexts in which to learn rarer symbols.

D. EXPERIMENTS: DETAILS

In this section we give more details of the experiments in §4.

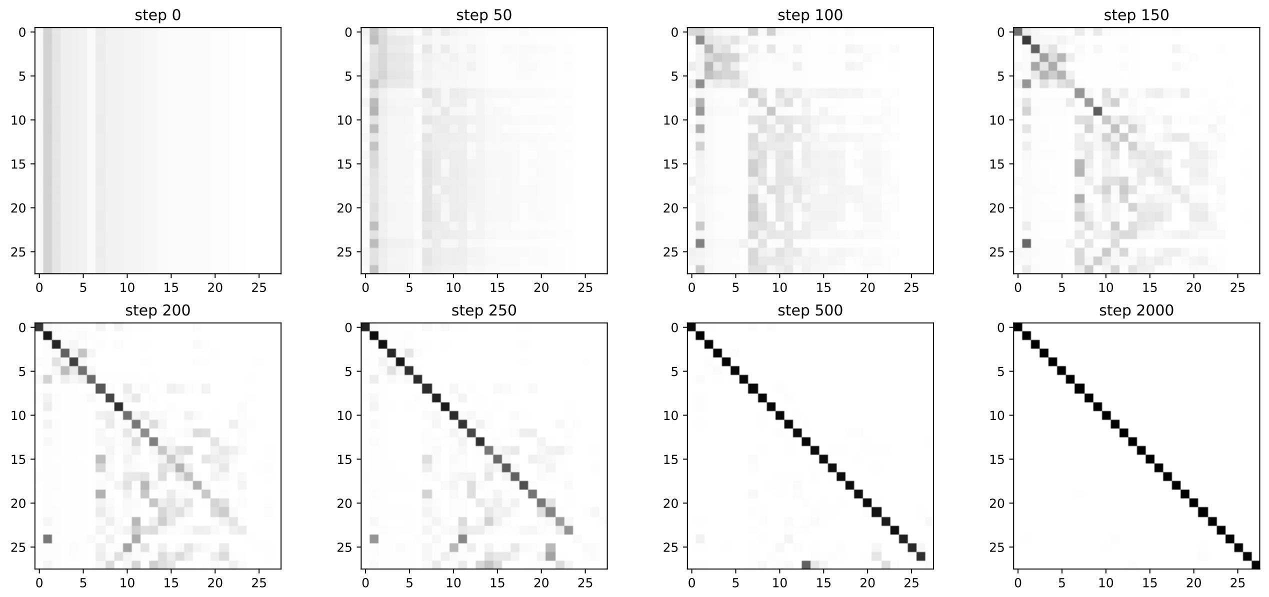

Figure 4. The matrix O learned after various numbers of training steps for a successful training run. By step 50, a tentative mapping of vowels to vowels and consonants to consonants has been learned. By step 100, the correspondence for the space character and a few of the most common individual vowels and consonants has been tentatively, and by step 150 clearly, learned. Clear knowledge of a few symbols opens up contexts for learning the association of other symbols based on their statistical properties, and learning progresses rapidly for less and less common symbols through steps 200 and 250. Finalizing the precisely correct association for very rare consonants such as z takes many steps.

D.1. Experimental setup: details

We use LibriTTS (Zen et al., 2019). The phoneme sequence for each utterance is obtained by forced alignment using the decoder graph of possible verbalizations and pronunciations of the text transcript, discarding any timing information. We limit training to utterances with grapheme and phoneme sequences of at most 96 tokens, yielding 166 hours of data. We randomly partition the training set, selecting a fraction γ of utterances as our paired speech–text examples and evenly splitting the rest into text-only and speech-only datasets, that is $\alpha = \beta = \frac{1-\gamma}{2}$. Output distribution temperature (described in §A.5) $T = 0.5$ is used during training. We pre-train (described in §A.5) for $100\,000$ steps (though far fewer steps typically suffice). Decoding is performed by stochastic sampling from $p_\lambda(y|x)$, also with $T = 0.5$. We compute *phoneme error rate (PER)* for a generative model with x a grapheme sequence and y a phoneme sequence, and *character error rate (CER)* for a generative model with x a phoneme sequence and y a grapheme sequence. To compute CER, each grapheme sequence is normalized by lowercasing and removing punctuation.

The prior, decoder and variational posterior are all modeled autoregressively. We use a recurrent neural net (RNN)-based architecture for the prior. The decoder and variational posterior are each parameterized as a seq2seq model with monotonic attention, similar to *listen, attend and spell (LAS)* (Chan et al., 2016) and Tacotron (Wang et al., 2017). To impose time locality on the decoder $p_\lambda(y|x)$ as discussed in §3, the input to the final RNN predicting the distribution over y_t consists only of glimpses of x chosen by the attention mechanism and does not directly include any information about $y_{<t}$. No such constraint is imposed on the variational posterior $q_\nu(x|y)$ to ensure it remains as flexible as possible. $p_\lambda(y|x)$ and $q_\nu(x|y)$ do not play symmetric roles. A variational posterior which ignores y incurs a large loss in (2) (the optimal solution is $q_\nu(x|y) = p_\lambda(x|y)$). However, a decoder $p_\lambda(y|x) = r(y)$ which ignores x incurs no loss in (1) for $\gamma = 0$. Empirically a flexible decoder destroys the performance (Table 1) as hypothesized (§3). Reproducibility details are given in §D.3.

D.2. Initialization

When approaching $\gamma = 0$ (no paired data), the model exhibited more sensitivity to random initializations and $\gamma = 0.0002$ (2 minutes of paired data) was the cutoff at which the model was never able to recover from poor local optima. Pre-training $p_\lambda(y|x)$ with a small amount of paired data (50 minutes) and then continuing training using only unpaired data gave metric

Module	Hyperparameters
One-hot	
Causal Conv1D	filters = 128, kernel size = 8, activation = ReLU
RNN w/ GRU cells	units = 128, zoneout probability = 0.01
Dropout	dropout rate = 0.05
Dense	units = 128, activation = ReLU

Table 3. Summary of the autoregressive prior $p_\lambda(x)$ architecture and hyperparameters.

Module	Hyperparameters
Input encoder	One-hot
	Conv1D: filters = 128, kernel size = 8, activation = ReLU
	Conv1D: filters = 128, kernel size = 8, activation = ReLU
Autoregressive decoder	Causal Conv1D filters = 64, kernel size = 12, activation = ReLU
	attention LSTM units = 64
	montonic GMM attention (Battenberg et al., 2020)
	→num components = 5, num heads = 1, units = 32
	→init offset bias = 1.0, init scale bias = 5.0
	decoder LSTM (units = 64)
	→input to decoder LSTM: only the attention glimpse

Table 4. Summary of the autoregressive decoder $p_\lambda(y|x)$ architecture and hyperparameters.

improvements similar to the results in Figure 1, suggesting that fully unpaired training ($\gamma = 0$) could be attainable with better initialization or optimization.

D.3. Reproducibility

Here we provide details on the model architecture and parameters, training hyperparameters and losses required to reproduce the results. The decoder and variational posterior are both simplified version of the Tacotron model (Wang et al., 2017) with output spectrogram predictor replaced with a categorical distribution. Tacotron has several open source implementations[§] and is straightforward to reproduce.

Prior parameterization: The prior $p_\lambda(x)$ is parameterized by an RNN-based autoregressive model, with a causal convolutional preprocessing of past samples. The detailed building blocks of this architecture is summarized in Table 3.

Decoder parameterization: We use a stack of two 1D convolution layers as input encoder, and a recurrent monotonic GMM attention model (Battenberg et al., 2020), to extract glimpses of the encoded input to feed to the decoder LSTM. When predicting y_t no receptive field to $y_{<t}$ is allowed. The details of this architecture is listed in Table 4. Note that a more powerful decoder with receptive field to past samples did not make a difference on prediction metrics of the supervised-only model. So, for simplicity, we kept the decoder consistent between the supervised-only and semi-supervised setups.

Variational posterior parameterization: The variational posterior $q_\nu(x|v)$ is very similar to the decoder, but slightly more powerful by allowing prediction of x_t to have receptive field to $x_{<t}$ via state of the attention LSTM, and also having a skip connection from input to decoder RNN. The details of this architecture is listed in Table 5.

Training hyperparameters We use separate Adam optimizers (Kingma & Ba, 2014) for generative parameters λ and variational parameters ν, and train the model for 300 000 steps. The generative optimizer learning rate is piecewise constant of values of $1e{-}3$, $5e{-}4$, $3e{-}4$, $1e{-}4$, $5e{-}5$, changing every 50 000 steps, and fixed after 200 000 steps. The variational optimizer uses fixed higher learing rate of $3e{-}3$ to keep the variational posterior up to date with respect to generative

[§]See `https://github.com/NVIDIA/tacotron2`, `https://github.com/Rayhane-mamah/Tacotron-2` and `https://github.com/keithito/tacotron`, for example.

Module	Hyperparameters
Input encoder	One-hot
	Conv1D: filters = 128, kernel size = 8, activation = ReLU
	Conv1D: filters = 128, kernel size = 8, activation = ReLU
Autoregressive posterior	Causal Conv1D filters = 64, kernel size = 12, activation = ReLU
	attention LSTM units = 64
	montonic GMM attention (Battenberg et al., 2020)
	$\rightarrow$num components = 5, num heads = 1, units = 64
	$\rightarrow$init offset bias = 1.0, init scale bias = 5.0
	decoder LSTM (units = 64)
	$\rightarrow$input to decoder LSTM: attention glimpse, attention RNN state, x_{t-1}

Table 5. Summary of the autoregressive variational posterior $q_\nu(x|v)$ architecture and hyperparameters.

parameters. Both optimizer apply gradient clipping.

Training losses: Section A.5 summarizes all the training losses which are simple interaction between sampling and log prob computations of the sequential prior, decoder and variational posterior distributions. Given the availability of these building blocks in open source community, here we only provide the novel proposed training objectives in code snippets. The generative model (λ) losses are summarized in Snippet 1, and the variational model (ν) losses are summarized in Snippet 2.

```
def generative_model_losses(x, y, x_observed, y_observed, xy_observed):
    """Computes generative model losses.
      - p_x ($p_\lambda(x)$), p_y_given_x ($p_\lambda(y|x)$) and q_x_given_y ($q_\nu(x|y)$) are sequence
        distribution abstractions with sample() and log_prob() APIs, where p_x
        is an unconditional RNN-based autoregressive model and p_y_given_x and
        q_x_given_y are Tacotron-like autoregressive attention-based seq2seq models.

    Args:
       x: A batch of x sequences.
       y: A batch of y sequences.
       x_observed: A [batch] boolean tensor indicating x-only observations.
       y_observed: A [batch] boolean tensor indicating y-only observations.
       xy_observed: A [batch] boolean tensor indicating x & y observations.
    Returns:
       a dict of losses tensors.
    """

    # Compute $\log p_\lambda(x)$.
    p_x_lp_data = p_x.log_prob(x)

    # Compute $\log p_\lambda(y|x)$.
    p_y_given_x_lp_data = p_y_given_x.log_prob(y, conditions={'x': x})

    # Compute $\log p_\lambda(x, y)$.
    p_xy_lp_data = p_x_lp_data + p_y_given_x_lp_data

    # Draw a sample $\hat{x} \sim q_\nu(x|y)$.
    x_inferred = q_x_given_y.sample(conditions={'y': y})

    # Compute $\log q_\nu(\hat{x}|y)$.
    q_x_given_y_lp_inferred = q_x_given_y.log_prob(x_inferred, conditions={'y': y})

    # Compute $\log p_\lambda(\hat{x})$.
    p_x_lp_inferred = p_x.log_prob(x_inferred)

    # Compute $\log p_\lambda(y|\hat{x})$.
    p_y_given_x_lp_inferred = p_y_given_x.log_prob(y, conditions={'x':x_inferred})

    # Compute $\log p_\lambda(\hat{x}, y)$.
    p_xy_lp_inferred = p_x_lp_inferred + p_y_given_x_lp_inferred

    # Compute ELBO: $\log p_\lambda(\hat{x}, y) - \log q_\nu(\hat{x}|y)$.
    p_y_elbo_data = p_xy_lp_inferred - q_x_given_y_lp_inferred

    p_xy_lp = mask(p_xy_lp_data, xy_observed) # paired x & y
    p_x_lp = mask(p_x_lp_data, x_observed) # x-only
    p_y_elbo = mask(p_y_elbo_data, y_observed) # y-only

    return {
       'neg_p_xy_lp': -p_xy_lp,
       'neg_p_x_lp': -p_x_lp,
       'neg_p_y_elbo': -p_y_elbo,
    }
```

Snippet 1. Code snippet computing losses of the generative model (see (1))

paired minutes (fraction)	dev		test	
	PER / %	CER / %	PER / %	CER / %
5 (γ=0.0005)	47.8	61.6	49.6	60.6
10 (γ=0.001)	33.2	46.9	34.2	45.7
20 (γ=0.002)	19.0	21.7	18.8	20.2
50 (γ=0.005)	12.1	14.6	11.0	13.5
100 (γ=0.01)	7.2	11.9	6.8	10.5
199 (γ=0.02)	5.0	9.6	5.6	10.0
498 (γ=0.05)	3.4	5.7	3.3	6.6
996 (γ=0.1)	2.8	4.5	2.5	3.9
9960 (γ=1)	0.9	2.1	1.3	1.7

Table 6. Phoneme and character error rates for utterance-level g2p and p2g at various paired supervision rates γ on LibriTTS dev and test sets for supervised-only approach.

paired minutes (fraction)	dev		test	
	PER / %	CER / %	PER / %	CER / %
5 (γ=0.0005)	11.7	25.6	9.4	9.5
10 (γ=0.001)	11.1	5.3	7.2	4.5
20 (γ=0.002)	6.3	4.7	5.6	4.2
50 (γ=0.005)	4.2	3.7	3.6	3.8
100 (γ=0.01)	3.6	3.3	3.5	3.2
199 (γ=0.02)	3.0	2.8	2.6	2.9
498 (γ=0.05)	3.0	3.1	2.3	3.6
996 (γ=0.1)	2.2	3.2	2.3	2.8

Table 7. Phoneme and character error rates for utterance-level g2p and p2g at various paired supervision rates γ on LibriTTS dev and test sets for the proposed semi-supervised generative modeling approach.

```
def variational_model_losses():
    """Computes variational model losses.
      - p_x (p_λ(x)), p_y_given_x (p_λ(y|x)) and q_x_given_y (q_ν(x|y)) are sequence
        distribution abstractions with sample() and log_prob() APIs, where p_x
        is an unconditional RNN-based autoregressive model and p_y_given_x and
        q_x_given_y are Tacotron-like autoregressive attention-based seq2seq models.

    Returns:
       a dict of losses tensors.
    """

    # Draw a sample x̂,ŷ ~ p_λ(x,y).
    x_gen = p_x.sample()
    y_gen = p_y_given_x.sample(conditions={'x': x_gen})

    # Compute log q_ν(x̂|ŷ).
    q_x_given_y_lp_gen = q_x_given_y.log_prob(x_gen, conditions={'y': y_gen})

    return {
      'neg_q_x_given_y_lp_gen': -q_x_given_y_lp_gen,
    }
```

Snippet 2. Code snippet computing losses of the variational model (see (2))

D.4. Experimental results

The numerical values of prediction errors for supervised-only and semi-supervised models are reported in Table 6 and Table 7 respectively.

E. RELATED WORK: DETAILS

Lately remarkable progress has been made in supervised ASR (Gulati et al., 2020; Han et al., 2020; Chan et al., 2021) and TTS (Wang et al., 2017; Shen et al., 2018; Ren et al., 2021) systems, powered by availability of massive parallel text and speech corpora, like LibriSpeech (Panayotov et al., 2015) and LibriTTS (Zen et al., 2019). Due to scarcity of such resources across all languages, there has been a great interest in leveraging non-parallel data (i.e., unspoken text and untranscribed speech) which are readily available at larger scales, without the need for manual transcription.

Toward this goal, self-supervision with various self-consistency training objectives has proven to be an effective way to pre-train speech encoder for ASR (e.g., CPC (Oord et al., 2018), wav2vec (Schneider et al., 2019), vq-wav2vec (Baevski et al., 2020a), wav2vec 2.0 (Baevski et al., 2020b), HuBERT (Hsu et al., 2021), W2v-BERT (Chung et al., 2021)), and text / phoneme encoder for TTS (e.g., (Hayashi et al., 2019), PnG BERT (Jia et al., 2021)). However these pre-trained models need to be fine-tuned with parallel data on the task of interest (e.g., paralinguistics (Shor et al., 2022), speaker verification (Chen et al., 2022a), or ASR (Hsu et al., 2021)), whereas semi-supervised learning allows extracting useful information from both speech-only and text-only datasets for ASR (Li et al., 2019; Chen et al., 2021a) and TTS (Chung et al., 2019). Recent developments in textless NLP models aim to uncover parts of the connection between text and speech from speech-only data (Lakhotia et al., 2021; Borsos et al., 2022).

Learning to associate the spoken and written language with little or no parallel data has been extensively explored but remains unsolved. There are three main veins of work. The first family of models use weight sharing in a multi-task speech / text representation learning setup to encourage alignment between the modalities in the encoded representation, which is always accompanied by some form of supervised objective on paired data to force that alignment (e.g., SpeechT5 (Ao et al., 2021), SLAM (Bapna et al., 2021), MAESTRO (Chen et al., 2022b)). The second family are primarily based on the back translation technique first adopted for machine translation (Sennrich et al., 2016) to incorporate monolingual data in a target language, by pairing them with automatic translation to the source language to generate synthetic paired data. This is closely related to speech chain theory that hypothesizes a connection between speech perception and production with a reinforcing feedback loop (Denes et al., 1993). This connection has motivated the joint training of ASR / TTS models in an auto-encoding setting on non-parallel data (Tjandra et al., 2017), with the limitation of not being able to the back-propagate to the ASR model when speech is auto-encoded, due to non-differentiable ASR textual outputs, which was later remedied by a straight-through gradient estimation (Tjandra et al., 2019). To avoid the mismatches between synthetic and real speech a similar cycle-consistency approach has been adopted on output of an ASR encoder (Hayashi et al., 2018; Hori et al., 2019). The use of pre-trained TTS as data augmentation (Rosenberg et al., 2019) or to guide self-supervised speech representation (Chen et al., 2021b) can also be considered variations of this technique. All of the models under this category are also trained with some amount of paired data. The third family of models is based on distribution matching, minimizing some divergence when mapping unpaired data to the other modality, including adversarially matching the phoneme distribution output by a speech encoder and phonemes derived from a text corpus (Liu et al., 2018; Baevski et al., 2021; Liu et al., 2022a) to build unsupervised ASR models. The tokens discovered by Baevski et al. (2021) have also been used as conditioning input to build an unsupervised TTS model (Liu et al., 2022b). The distribution matching family is the most principled approach and is the only one that has some success in zero paired data setup to the best of our knowledge.