DOI: 10.17469/O2111AISV000020

VINCENZO SCOTTI, ROBERTO TEDESCO

On the Role of Dialogue Context in Predicting Speaking Style

Text-to-Speech (TTS) synthesis is a problem almost as old as Natural Language Processing (NLP). The focus of this problem is on creating tools capable of generating a voice uttering a given text. Deep learning-powered solutions try to go beyond mere speech generation from the text: newer models try to factorise the probability predicted by these generative models to condition it on various aspects: speaker's voice, speaking style, or prosody. In this work, we focused on predicting the speaking style from the given text inside a conversation, for the application to conversational agents and chatbots. To this end, we developed and trained a neural network module working as a connector between the textual component of a chatbot (i.e., the neural language model for dialogue understanding and generation) and the speech synthesis component of a chatbot (i.e., the neural TTS synthesis model with speaking style conditioning).

Keywords: Natural Language Processing, Deep learning, Expressive speech synthesis, Global Style Token.

1. *Introduction*

Large deep learning-based generative models have become a standard tool for Natural Language Processing (NLP). They can be used for text understanding and generation (Raffel, Shazeer, Roberts, Lee, Narang, Matena, Zhou, Li, Liu, 2020; Brown, Mann, Ryder, Subbiah, Kaplan, Dhariwal Neelakantan Shyam Sastry, Askell, Agarwal, Herbert-voss, Krueger Henighan, Child, Ramesh, Ziegler, Wu, Winter, Hesse, Chen, Sigler, Litwin, Gray, Chess, Clark, Berner, Mccandlish, Radford, Sutskever, Amodei., 2020), as well as for speech recognition and synthesis (Tan, Qin, Soong, Liu, 2021; Radford, Kim, Xu, Brockman, Mcleavey, Sutskever, 2022). Recent advances have shown how the text-based models can be adopted to build incredibly versatile chatbots (i.e., conversational agents, OpenAI, 2022; Google, 2023).

In this work, we are interested in developing a solution to be adopted by empathetic chatbots. These chatbots simulate empathy to the end of being perceived as human like by the users. They find applicability in applications like automatic therapy, counselling, and coaching. Research on empathetic chatbots showed that embodying a conversational agent though voice or an avatar improves the pro-social attitude of the user towards the agent, helping the human-like perception (Paiva, Leite, Boukricha, Wachsmuth, 2017).

For this scope, we developed our Dialogue Global Style Token (DGST) module. This module builds on top of state-of-the-art approaches for Expressive Speech Synthesis that use a latent code to condition the generated speech towards a specific

style. This latent code is called Global Style Token (GST, Wang, Stanton, Zhang, Skerry-Ryan, Battenberg, Shor, Xiao, Jia, Ren, Saurous, 2018).

We divide the rest of this paper into the following sections. In § 2 we present the state of the art on neural network-based TTS synthesis. In § 3 we describe the methodology we used to infer speaking style from text using dialogue language models in the DGST model. In § 4 we describe the experimental approach we followed to search for the best DGST configuration to predict speaking style from text, the dialogue data we used in our experiments and the results obtained from the different configurations. Finally, in § 5 we summarise our contribution and suggest possible future extensions.

2. *State of the art on neural TTS*

In this section we present the current state of the art for TTS synthesis, with emphasis on the techniques for expressive speech synthesis. Neural network-based TTS use a pipeline divided into two main components: acoustic model (also called spectrogram predictor) and vocoder (Jurafsky, Martin, 2022; Tan *et al.*, 2021). The former component takes care of converting a sequence of graphemes[1] or phonemes[2] into a (Mel-) spectrogram; the latter component takes care of converting the Mel-spectrogram into a raw waveform, concluding the synthesis process.

2.1 Neural network-based speech synthesis

The acoustic models for Mel-spectrogram prediction are commonly built using a Seq2Seq transducer architecture (referring to Fig. 1, the bordeaux and green parts are respectively the input phoneme/grapheme sequence and the output (Mel-) spectrogram). These models are mainly composed by an encoder of the input text and a decoder that generates the output speech. Often the architecture is extended to allow for some form of conditioning on the generated speech. These networks are trained to minimise the mean squared error (MSE) between the reference and predicted spectrogram.

The encoder takes as input a sequence of graphemes or phonemes. Some models even allow using both encodings (Valle, Li, Prenger, Catanzaro, 2020). The symbols in the input sequence are considered orthogonal symbols and are used to fetch embedding vectors that are processed by the encoder. The encoder usually leverages a bi-directional approach to find a latent representation of the input that can be used to condition the decoder to generate the correct spectrogram. The sequence of latent vectors (one for each phoneme or grapheme in the input) in output from the

[1] In this context we define a *grapheme* as the "the smallest meaningful contrastive unit in a writing system". In other words, it is a written symbol that allows distinguishing a *minimal pair* (e.g., the graphemes h and m allow to distinguish the two words in the minimal pair [house, mouse]).

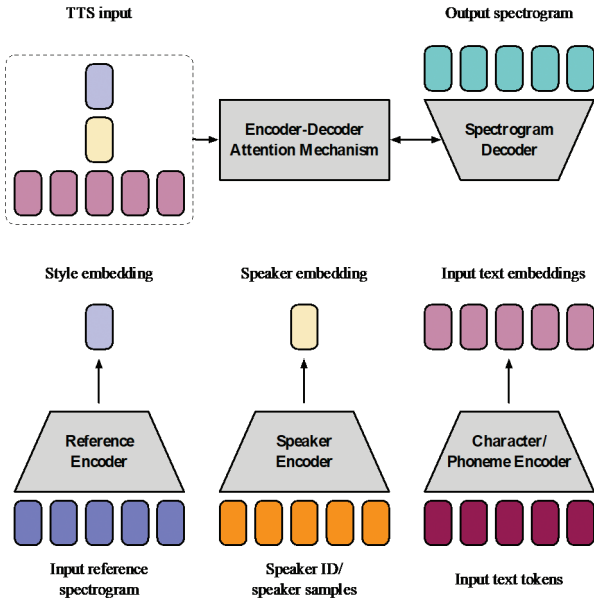
[2] In this context we define a *phoneme* as any of the perceptually distinct units of sound in a specified language that distinguishes one word from another, for example, p, b, d, and t in the English words pad, pat, bad, and bat.

encoder is used a reference from the decoder to align the generated speech to the input text to utter (see Fig. 1).

Between the encoder and decoder, there is an attention mechanism that allows learning implicitly the alignment between input and output. The activation of the attention can be retained and used to visualise this alignment.

The decoder usually generates the output autoregressively, consuming the latest generated slice of spectrogram (thus a spectrum) to output the following one. During training this approach allows guiding the generation process providing as input always the correct spectrum, taking it from the reference and speeding up the training process. At inference time, however, the decoder works completely on its own.

Figure 1 - *Example of encoder-decoder TTS synthesis with speaker and speaking style conditioning*



Tacotron (Shen, Pang, Weiss, Schuster, Jaitly, Yang, Chen, Zhang, Wang, Skerry-Ryan, Saurous, Agiomyrgiannakis, Wu, 2018; Wang, Skerry-Ryan, Stanton, Wu, Weiss, Jaitly, Yang, Xiao, Chen, Bengio, Le, Agiomyrgiannakis, Saurous Clark, Saurous, 2017) and DeepVoice (Ping, Peng, Gibiansky, Arik, Kannan, Narang, Raiman, Miller, 2018) are two examples of encoder-decoder models. The former uses recurrent neural networks as the main component while the latter uses convolutional networks. These two architectures have a separate module to predict the stopping point of the generation process.

FastSpeech (Ren, Ruan, Tan, Qin, Zhao, Zhao, Liu, 2019) represents instead a very different example of TTS. It uses a feed-forward architecture going from the input text to the output speech in a single module. The main architecture is based on that of an encoder-only transformer, processing in parallel the input and

generating the output. In this case, there are some additional modules to predict the phone duration. The encoder takes as input a sequence of graphemes or phonemes. Some models even allow using both encodings. The symbols in the input sequence are considered orthogonal symbols and are used to fetch embedding vectors that are processed by the encoder. The encoder usually leverages a bi-directional approach to find a latent representation of the input that can be used to condition the decoder to generate the correct spectrogram.

To complete the speech synthesis pipeline, a vocoder is necessary. The vocoder converts the output of the spectrogram predictor into an intelligible waveform. Neural vocoders have become a fundamental module of TTS models; they are necessary to synthesise a speech as clearly as possible (Jurafsky, Martin, 2022). Compared to the previous approach, the Griffin-Lim algorithm (Griffin, Lim, 1983), neural vocoders yield audio with fewer artefacts and higher quality.

Available implementations are based on (dilated) convolutional neural networks (e.g., WaveNet, Oord, Dieleman, Zen, Simonyan, Vinyals, Graves, Kalch-Brenner, Senior, Kavukcuoglu, 2016; WaveGlow, Prenger, Valle, Catanzaro, 2019; or MelGAN, Kumar *et al.*, 2019; Yang *et al.*, 2021) or recurrent neural networks (e.g., WaveRNN, Kalchbrenner, Elsen, Simonyan, Noury, Casagrande, Lockhart, Stimberg, Oord, Dieleman, Kavukcuogl., 2018).

There are two approaches to training these models. They can be either trained to work directly on raw waveforms (Oord *et al.*, 2016) or Mel-spectrograms (Yang *et al.*, 2021).

2.2 Expressive speech synthesis

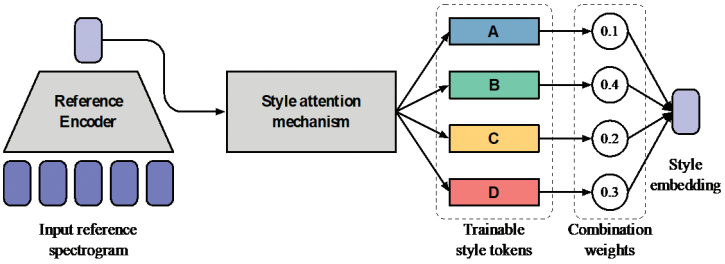
Besides the conditioning on the synthesised speech given by the input text (what to say), there are multiple research lines focused on controlling further aspects of the synthesised speech through additional information. These additional aspects can be categorised into two groups: speaker (or timbre, i.e., who is speaking) and prosody (or style, i.e., how to speak) (Tan *et al.*, 2021). These two aspects are completely orthogonal and can be combined (Skerry-Ryan, Battenberg, Xiao, Wang, Stanton, Shor, Weiss, Clark, Saurous, 2018).

Speaker control allows disentangling the uttered content from the speaker's voice, making the overall TTS model more reusable. The idea is to provide additional information on the speaker to the TTS. This approach allows leveraging multi-speaker data sets, while, previously, neural TTS used to be trained on single-speaker data sets (Wang *et al.*, 2017). To implement this kind of conditioning, it is sufficient to concatenate the hidden features extracted from a speaker recognition network to the hidden representation of the input text, as it was done for Tacotron 2 (Jia, Zhang, Weiss, Wang, Shen, Ren, Chen, Nguyen, Pang, Lopez-moreno, Wu., 2018; Favaro, Sbattella, Tedesco, Scotti., 2021; Favaro, Sbattella, Tedesco, Scotti, 2023) and DeepVoice 3 (Ping *et al.*, 2018).

In this paper we refer to the subtractive definition of prosody from Skerry-Ryan *et al.* (2018), where they consider prosody as the variation in speech signals

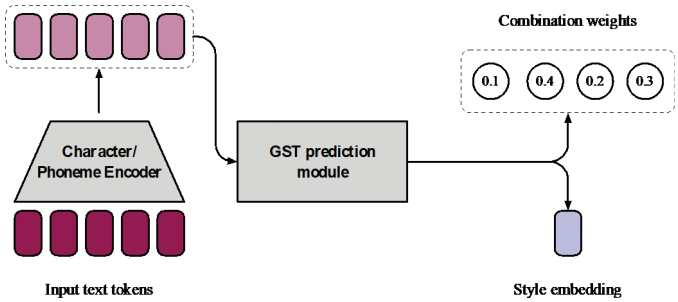
that remains after accounting for variation due to phonetics, speaker identity, and channel effects. In recent work on expressive TTS, prosody and style are used interchangeably. Prosodic control covers all the aspects concerning intonation, stress, and rhythm, which characterise how a sentence is uttered. These aspects also influence the emotion perceived by the listener. A reference step towards this kind of control on the speaking style is represented by the GSTs (Wang *et al.*, 2018). Instead of explicitly modelling the aspect characterising the prosody, this model uses unsupervised style representations learnt alongside the synthesis model. In the original implementation, a Tacotron 2 model was extended with a separate encoder that extracted a vector representing the so-called style embedding; this vector is concatenated to the hidden representation of the input text to provide the decoder with the style information (Wang *et al.*, 2018). We represented the GST computation process in Fig. 2. Notice that this kind of control works at a high level: the specific changes connected to a given style are learnt and implemented by the spectrogram generator.

Figure 2 - *Style embedding extraction using GSTs and combination weights from reference audio*



GSTs are designed to be learned from audio and used by extracting them from a reference clip. This would require having an available database of reference clips, which is not always a viable solution. Recently, approaches have been proposed to predict the GST from the raw text (Stanton, Wang, Skerry-Ryan, 2018) and solve this issue (see Fig. 3). There are two possible approaches: either predict the GST combination weights or directly predict the style embedding. The former is like a classification problem, where the output is a distribution over the components of the actual style embedding, the latter is instead a regression problem, where the style embedding is the target variable.

Figure 3 - *Style embedding and GST weights inference using from input text phonemes*



3. *Methodology*

In this section, we present DGST, explaining how it predicts the speaking style and how it is trained to predict such speaking style. This model revolves around the concept of style embedding, a (latent) vector representation of style in a dense continuous space.

3.1 Overview

In GST-powered speech synthesis modules, there is a sub-module in the network generating a latent vector representation of the speaking style, used to condition the speech synthesis (Valle *et al.*, 2020; Wang *et al.*, 2018). This latent vector is computed as the weighted combination of parameter vectors internal to the network. The weights of the combination are computed with the attention mechanism: a reference audio clip is processed into a single vector used as a query for attention, while the TTS provides the vectors to be used to compute keys and values. Some models, like Mellotron, adopt a multi-head attention approach, producing multiple output vectors (as many as the attention heads) and combining them (in the case of Mellotron, the vectors are simply concatenated).

Usually, the style is extracted from a reference speech clip, which may not be available at run time. Previous solutions for GST prediction from text proposed by Stanton *et al.* (2018), focused on predicting the style simply from the text to utter, predicting either the prosody embedding directly or predicting the combination weights to be passed to the attention of the style encoder in the TTS. They used as input the encoding of the text to utter realised by the TTS, which is usually computed from the phonemes.

The module we build, DGST, proposes to focus on the dialogue domain, also including information from the context of the conversation. We leverage the hidden representations of a language model, which are usually very informative and yield impressive results on many regression and classification tasks.

3.2 DGST model

The model we propose combines two tasks together: a regression task to predict the style embedding and a classification task to predict the style combination weights (the combination weights are a distribution of probability over the available style tokens, i.e., the latent codes).

DGST takes as input a 1D feature map (i.e., a matrix) $H \in R^{t \times d}$ (t is the number of elements in the input sequence and d is the number of dimensions of the vectors encoding a single sequence element) encoding the text the TTS needs to utter. Each column vector in this feature map is an embedding computed by a deep learning-based language model.

DGST learns two output functions $f_{GSTe}(\boldsymbol{H}) = \hat{\boldsymbol{y}}_{GSTe} \in \mathbb{R}^{d_e}$ (d_e is the number of dimensions of a style embedding) and $f_{GSTw}(\boldsymbol{H}) = \hat{\boldsymbol{Y}}_{GSTw} \in \mathbb{R}^{h \times d_h}$ (h is the number of attention heads and d_h is the number of dimensions of a single attention head), both use as input the 1D feature map. The former tackles the problem of predicting directly the style embedding as if it was a regression problem. The latter tackles the problem of predicting the combination weights of style embeddings (note that we are considering a generic case where the GST module uses h different heads for the combinations weights on the style tokens).

What characterises DGST is the choice of how the input features are computed. We considered three possible approaches:

1. The input embeddings are computed from the current turn to utter in the dialogue and nothing else, as in Fig. 4. The model ignores the dialogue context and uses only information from the current response.
2. The input embeddings are computed from the current turn to utter in the dialogue and the preceding turn, but only the embeddings of the current turn are used by DGST, as in Fig. 5. Since deep learning-based language model exploit context to compute the hidden representations, even if we use only the embeddings from the last turn, they include information coming from the context.
3. The input embeddings are computed from the current turn to utter in the dialogue and the preceding turn, and all the embeddings are used by DGST, as in Fig. 6. In this case we are adding explicitly the information from the embeddings of previous turns.

DGST works entirely with latent representations, from input to output. It predicts the most probable style embedding or embedding combination weights from the contextual latent representation yield by a language model. Given these latent space interconnections, DGST is bound to the language model and expressive TTS used to compute its inputs during training. These modules cannot be changed or DGST would have to be re-trained.

Figure 4 - *Style embedding and GST weights inference from dialogue response contextual embeddings*

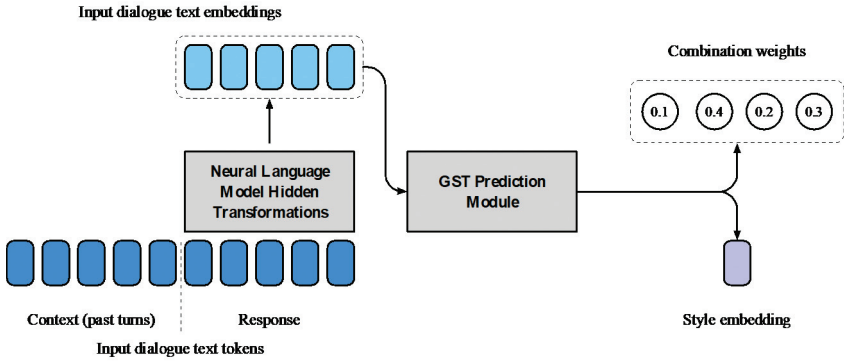


Figure 5 - *Style embedding and GST weights inference from dialogue response contextual embeddings (computed using also dialogue context)*

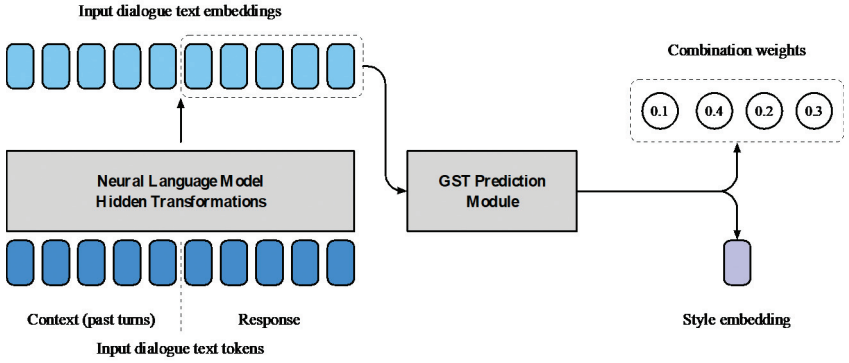
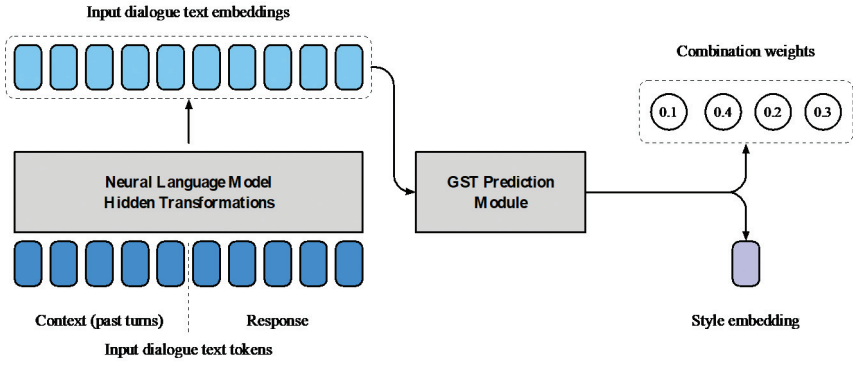


Figure 6 - *Style embedding and GST weights inference from dialogue context and response contextual embeddings*



3.3 Training approach

To train this network we employ two external models: a dialogue language model and a TTS with GST conditioning. The Dialogue language model is used to extract the contextual embeddings H that feed our model (i.e., the input), while the GST module of the TTS is used to extract the targets $\boldsymbol{y}_{GSTe} \in \mathbb{R}^{d_e}$ and $\boldsymbol{Y}_{GSTw} \in \mathbb{R}^{h \times d_h}$ from audio clips containing the speech uttering the response processed by the dialogue language model.

The overall loss function $\mathcal{L}_{DGST}(\cdot)$ is described in (1).

$$(1)\ \mathcal{L}_{DGST}(\boldsymbol{H}, \boldsymbol{y}_{GSTe}, \boldsymbol{Y}_{GSTw}) = \mathcal{L}_{DGSTe}(\boldsymbol{H}, \boldsymbol{y}_{GSTe}) + \mathcal{L}_{DGSTw}(\boldsymbol{H}, \boldsymbol{Y}_{GSTw})$$

It is the sum of two losses, one for the embedding prediction head $\mathcal{L}_{DGSTe}(\cdot)$ and one for the combination weights prediction head $\mathcal{L}_{DGSTw}(\cdot)$. In the reference work by Stanton *et al.* (2018), they used the L1 norm loss for both heads, here instead, we distinguish between the two tasks of the heads.

Given that the embedding prediction head behaves as a regression model (i.e., a model predicting a continuously valued output, which is the embedding in our case), we use the mean squared error between the embedding prediction and the target embedding as loss function, as described by (2).

$$(2)\ \mathcal{L}_{DGSTe}(\boldsymbol{H}, \boldsymbol{y}_{GSTe}) = \frac{1}{d_e} \| \boldsymbol{y}_{GSTe} - f_{DGSTe}(\boldsymbol{H}) \|_2^2$$

Since the combination weights can be seen as a discrete distribution, we compute the loss of the combination weights prediction head as the distance between the reference and target distributions using the Kullback–Leibler (KL) Divergence. The KL Divergence $D_{KL}\,[P \parallel Q]$ of P from Q is a measure of how the probability distribution P is different from a reference probability distribution Q. We computed the average KL Divergence between the target weights and the predicted combination weights for each head. The KL Divergence is computed head-wise, and then we averaged the score among the heads as in (3).

$$(3)\ \mathcal{L}_{DGSTw}(\boldsymbol{H}, \boldsymbol{Y}_{GSTe}) = \frac{1}{h} \sum_{i=1}^{h} D_{KL}[\boldsymbol{Y}_{GSTw,i} \parallel f_{DGSTw}(\boldsymbol{H})_i]$$

4. *Experiments and results*

In this section we present the training and evaluation approaches we adopted with DGST (including the details on the experiments configurations). Subsequently we analyse and comment the obtained results.

4.1 Experimental approach

We trained multiple versions of DGST varying the underlying language model to extract the input features and varying how to extract such features. In any case, we trained for the prediction of both GST embedding and combination weights using a single model, exploiting the same hidden representation.

Concerning the language models, to observe the effects of their complexity and size, we evaluated three versions of GPT-2 fine-tuned on large dialogue corpora. To this end, we resorted to DialoGPT (Zhang, Sun, Galley, Chen, Brockett, Gao, Gao, Liu, Dolan, 2020), using the 117M, 345M and 762M parameter versions. Additionally, we repeated the same training steps using Therapy-DLDLM (Scotti, 2023), a custom fine-tuning of GPT-2 (762M parameter version) for dialogue that uses a discrete latent representation to encode dialogue turn. We choose this additional model to see whether exploiting its feature vectors (that can compress entire dialogue turns) would reach the same or comparable results of a more generic model pre-trained on larger data sets (Therapy-DLDLM should be the language model we are interested in using for the empathetic chatbot we are willing to provide with expressive speech capabilities).

We trained different models adopting all the feature extraction approaches presented in § 3. We used all the language models' hidden transformations to extract the sequence of embeddings from the response only, from the context and the response keeping only the hidden vectors of the response, and from the context and the response keeping all hidden vectors.

The actual DGST model is a single layer of Gated Recurrent Units (GRUs, Cho, Van Merrienboer, Gülçehre, C, Bahdanau, Bougares, Schwenk, Bengio., 2014) to process the input sequence followed by two alternative linear projections (one per target). Before the GRU layer, we applied dropout with a 10 % probability to regularise the model. We found that this configuration was sufficient to obtain good results.

We trained all models with a constant learning rate of $\eta = 5 \cdot 10^{-4}$ and a batch size of 32 elements. During training, we found that within 5 epochs, all models would reach a plateau on the validation scores. We used AdamW (Loshchilov, Hutter, 2019) as optimiser.

The main evaluation we conducted on this model is based on the training metrics. We compare the final test values, considering all the model variants we considered to understand the role of model complexity and dialogue context when predicting the GST embedding or the GST combination weights from the text.

To have statistical evidence that the results of the best performing model variant were the best, we run a t-test on these results. We compared the scores achieved by the best model with those of all the other variants, computing the p-value for statistical significance with the confidence level $\alpha = 0.05$. The null hypothesis was that the two populations had same mean and with a p-value $< \alpha$ we could reject that hypothesis. We run the test separately on KL-Divergence scores and MSE scores, in this way we could compare separately the combination weights prediction and the embedding prediction.

Additionally, we performed an evaluation using the spectrogram generator. We compared the Mel-spectrogram synthesised by the TTS using the reference clip and the GST embedding or the GST combination weights predicted by DGST using the best-performing configuration.

4.2 Data

We conducted the training and evaluation session of DGST on the IEMOCAP corpus (Busso, Bulut, Lee, Kazemzadeh, Mower, Ki Chang, Lee, Narayanan, 2008). IEMOCAP is a multimodal corpus for dialogue. This data set offers scripted and spontaneous dialogues in English displaying different emotions that result in a wide variety of speaking styles.

Our objective is to exploit this rich variety of speaking style characterising the corpus. The neural network implementing the DGST module should learn to choose the appropriate speaking style given the text to utter (i.e., the latest response in a dialogue) and, possibly, the conversational context (i.e., conversation history). Thanks to the audio recordings, we can use a pre-trained speaking style encoder coming from a TTS model to compute the targets to predict from input. In this work we used the style encoder of a Mellotron TTS model[3].

Table 1 - *IEMOCAP data set main statistics*

	Training	*Validation*	*Test*
Dialogues	84	29	38
Utterances	5754	1818	2373

In Tab. 1 we reported the main statistics on the corpus. Differently from common benchmark for chatbots, IEMOCAP does not contain many dialogues; however, the task is simpler than learning how to respond in a dialogue. Moreover, the individual conversations are longer. This should help the model learn long term dependencies.

4.3 Results and comments

We report the results on the target metrics for the two DGST approaches in Fig. 7 and Fig. 8. The scores show interesting behaviour.

From the results on DialoGPT it seems that the size of the language model leads to overfitting and that the 345M parameters version reaches the best performances. Moreover, the results seem to indicate that using the information from the context negatively affects the performances with respect to processing only the response. However, the results obtained using the features computed by Therapy-DLDLM highlight a completely different result.

We achieved the best results using the most complex language model (Therapy-DLDLM is built from a 762M parameters GPT-2) and processing with DGST the embeddings of the response computed also using the context. We hypothesise that the hierarchical nature of the variational model in Therapy-DLDLM enforces the

[3] We used a custom wrapper (link: https://github.com/vincenzo-scotti/tts_mellotron_api) over the original Mellotron TTS model (link: https://github.com/NVIDIA/mellotron); the models we used takes as input a mixture of phonemes (when the word to utter belongs to a vocabulary of pre-encoded words) and graphemes (when the word to utter is out of the reference vocabulary).

latent representation to extract a contextual embedding better summarising the sequence. Another possible explanation is in the quality of data, since DialoGPT was trained on a larger data set that was not manually curated as those used for Therapy-DLDLM; so, possibly, to learn a good model from these data sets a larger number of samples may be required.

Figure 7 - *Test set MSE on style embedding reconstruction using different approaches for input embeddings and different language models*

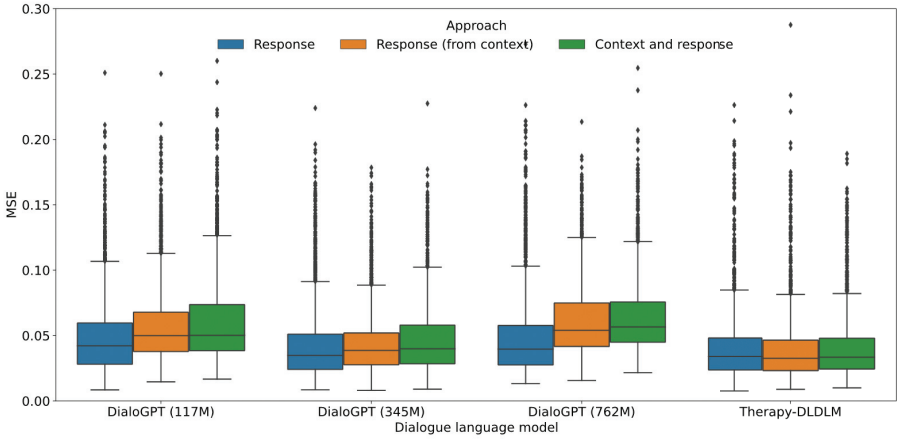


Figure 8 - *Test set KL-Divergence on GST weights distribution reconstruction using different approaches for input embeddings and different language models*

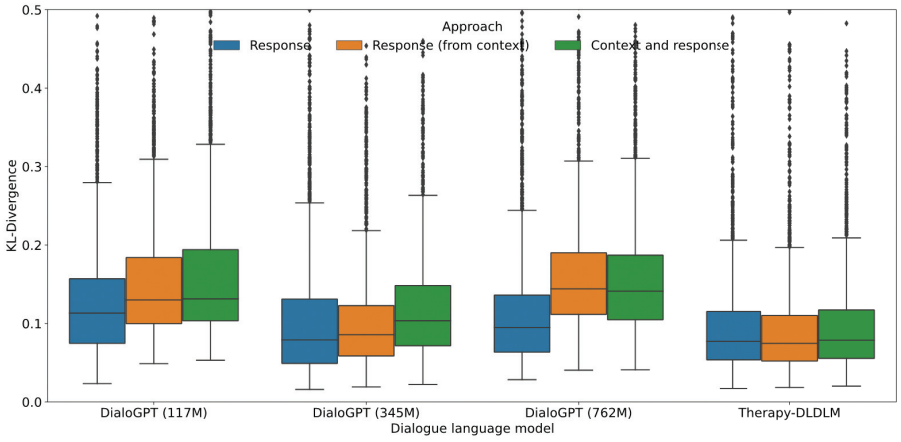


Table 2 - *T-test results comparing the best performing DGST model (DGST with Therapy-DLDLM contextual embeddings of dialogue response, computed also using the dialogue context) with all the other DGST versions*

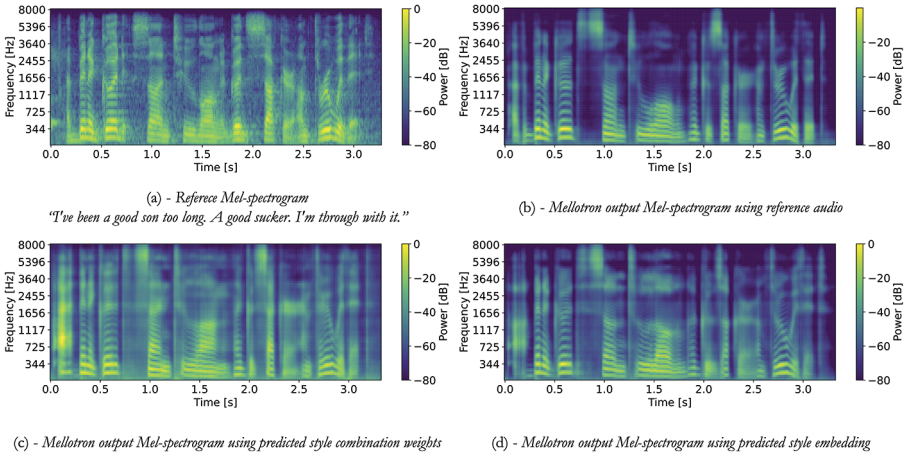
Approach	*p-value*			
	DialoGPT (117M)	*DialoGPT (345M)*	*DialoGPT (762M)*	*Therapy-DLDLM*
	MSE			
Response	1.777×10^{-34}	2.941×10^{-7}	4.707×10^{-33}	0.076
Response (from context)	5.310×10^{-114}	7.236×10^{-10}	3.015×10^{-172}	-
Context and response	1.635×10^{-136}	8.380×10^{-25}	4.181×10^{-203}	0.150
	KL-Divergence			
Response	1.218×10^{-74}	1.920×10^{-13}	1.202×10^{-29}	0.012
Response (from context)	1.361×10^{-180}	1.466×10^{-8}	3.930×10^{-233}	-
Context and response	9.014×10^{-193}	7.022×10^{-49}	4.401×10^{-208}	0.005

In Tab. 2 we report the p-values from the t-tests comparing the DGST model using Therapy-DLDLM transformer with the "response from context" encoding approach, which was the best-performing model on both approaches. In all tests, we set $\alpha = 0.05$ and, thus, a confidence level of 95 %. If compared to the models based on DialoGPT, DGST based on Therapy-DLDLM using the "response from context" not only performs better than the other models but has evidence that the difference in the results is statistically significant. When we compare the results of DGST with the same Therapy-DLDLM and the different encoding approaches, we can see that the results are statistically significant only on the KL-Divergence score, and thus the models predicting the GST combination weights. Differently, on the MSE scores (the models directly predicting the GST embeddings), despite we get better results, we don't have the same statistical evidence supporting the difference with the other encoding approaches.

To get an idea of the actual effect of the predicted GSTs on the synthesis, we visualised the generated Mel-spectrograms. We report an example in Fig. 9 comparing the original spectrogram with the one generated by Mellotron using the original audio clip for the style input and the two generated spectrograms using Mellotron and the GST predicted using the two approaches proposed with DGST.

The prediction using the GST embeddings has a noisier spectrogram, with an overall higher average power, as a result, the image of the Mel-spectrogram appears brighter and more like that of the source audio clip. The prediction using the GST combination weights is less noisy and is closer to the one generated using the original clip. In both cases, the spectrogram does not appear distorted and is not much different from the one generated using the two audio clips (look for example at the harmonics). This similarity indicates that both prediction approaches (either GST embeddings or GST combination weights) may be equally valid.

Figure 9 - *Comparison of reference Mel-spectrogram with synthesised Mel-spectrogram using Mellotron*



(a) - *Reference Mel-spectrogram*
"I've been a good son too long. A good sucker. I'm through with it."

(b) - *Mellotron output Mel-spectrogram using reference audio*

(c) - *Mellotron output Mel-spectrogram using predicted style combination weights*

(d) - *Mellotron output Mel-spectrogram using predicted style embedding*

5. *Conclusion*

In this paper we present a methodology to train a model to predict speaking style from text: DGST. DGST showed to be a valid tool to bridge text and speech, allowing control of the expressiveness of the speech by learning from human examples during conversations. The interesting finding is in the input features and the approach to extracting such features.

The best performances in terms of GST prediction, either considering GST embeddings or considering GST combination weights, are reached using the contextual embeddings yield by Therapy-DLDLM. However, we got statistical evidence on the results only when comparing models predicting the GST combination weights and not the model predicting the GST embeddings. Moreover, Therapy-DLDLM compares better than the same size and same architecture models trained on larger data sets (namely DialoGPT). We hypothesise this is due to the hierarchical nature of the variational model. Due to how Therapy-DLDLM processes the dialogue, the hidden representation of text computed by the model is forced to summarise the entire sequence, helping the computation of GSTs.

Including dialogue context seems to help the recognition of the GST to use for the response. However, rather than processing with the DGST module both context and response, processing only the embeddings of the response, encoded also using the context information, yields empirically the best results. We suppose that this is a consequence of using GRU for sequence modelling, which despite being a powerful sequence modelling tool, as other recurrent networks fails when the context gets too large. Nevertheless, using GRU allowed impressive results without resorting to the fine-tuning of the transformer dialogue model, which requires more computational effort and may need more samples to be correctly fine-tuned.

The next step we are considering for this kind of modules to connect speech and text is to drop the latent representation and start using explicit names to address the desired speaking style. Similarly to current solutions for image generation from text, where it is possible to ask for the desired style of the image (e.g., photorealistic image or oil painting), we are interested in asking through a textual prompt for the speaking style (e.g., loudly or mumbling). A solution of this kind would make the model easier to interpret and use.

Appendix

For transparency and reproducibility, we release the developed source code via GitHub at the following link: https://github.com/vincenzo-scotti/dialogue_gst

References

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., & Lin, H., (a cura di), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 6-12 December 2020, virtual.

Busso, C., Bulut, M., Lee, C., Kazemzadeh, A., Mower, E., Kim. S., Chang, J.N., Lee, S., and Narayanan, S.S., (2008). IEMOCAP: interactive emotional dyadic motion capture database. *Lang. Resour. Evaluation*, 42(4): 335–359.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Moschitti, A., Pang, B., Daelemans, *W., Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, 25-29 October 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 1724–1734. ACL. doi: 10.3115/V1/D14-1179.

Google, 2023, Introducing BARD. URL: https://blog.google/technology/ai/bard-google-ai-search-updates/

Jia, Y., Zhang, Y., Weiss, R.J., Wang, Q., Shen, J., Ren, F., Chen, Z., Nguyen, P., Pang, R., Lopez-Moreno, I., Wu, Y., (2018). Transfer learning from speaker verification to multispeaker text-to-speech synthesis. A cura di Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi,N., Garnett, R., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, Montréal, Canada, 3-8 December 2018, 4485–4495.

Jurafsky, D., Martin, J.H., (2009). *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*, 2nd Edition. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International.

Favaro, A., Sbattella, L., Tedesco, R., and Scotti, V., (2021). Itacotron 2: Transfering english speech synthesis architectures and speech features to italian. In Abbas, M., Alhakim Freihat, A., (a cura di), *4th International Conference on Natural Language and Speech Processing*, Trento, Italy, 12-13 November 2021, 81–86. URL https://aclanthology.org/2021.icnlsp-1.10

Favaro, A., Sbattella, L., Tedesco, R., & Scotti, V., (2023). ITAcotron 2: The Power of Transfer Learning in Expressive TTS Synthesis, Springer International Publishing, Cham, 1-20. doi: 10.1007/978-3-031-11035-1_1.

Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg, F., Oord, V.D., A., Dieleman, S., Kavukcuogl, K., (2018). Efficient neural audio synthesis. In. Dy, J.G., Krause, A., *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Stockholmsmässan, Stockholm, Sweden, 10-15 July 2018, vol. 80 of Proceedings of Machine Learning Research, 2415–2424.

Loshchilov, I., Hutter, F., (2019). Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019,* New Orleans, LA, USA, 6-9 May 2019.

OpenAI, 2022, Introducing ChatGPT, URL: https://openai.com/blog/chatgpt

Paiva, A., Leite, I., Boukricha, H., Wachsmuth, I., (2017). Empathy in virtual agents and robots: A survey. *ACM Trans. Interact. Intell. Syst.*, 7(3):11:1–11:4. doi: 10.1145/2912150.

Picard, W.P., (1997). *Affective computing*. MIT press.

Ping, W., Peng, K., Gibiansky, A., Arik, S.Ö., Kannan, A., Narang, S., Raiman, J., Miller, J., (2018). Deep voice 3: Scaling text-to-speech with convolutional sequence learning. In *6th International Conference on Learning Representations, ICLR 2018*, Vancouver, BC, Canada, 30 April – 3 May 2018, Conference Track Proceedings. OpenReview.net.

Prenger, R., Valle, R., Catanzaro, B., 2019, Waveglow: A flow-based generative network for speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019*, Brighton, United Kingdom, 12-17 May 12019, 3617–3621.

Radford, A., Kim, J.W., Xu, T., Brockman, G., McLeavey, C., Sutskever, I., (2022). Robust Speech Recognition via Large-Scale Weak Supervision. In: *CoRR* abs/2212.04356. DOI: 10.48550/ARXIV.2212.04356.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. In *J. Mach. Learn. Res.*, 21:140:1–140:67.

Ren, Y., Ruan, Y., Tan, X., Qin, T., Zhao, S., Zhao, Z., Liu, T., (2019). Fastspeech: Fast, robust and controllable text to speech. In Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R., (a cura di), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, Vancouver, BC, Canada 8-14 December 2019, 3165–3174.

Scotti, V., (2023). *Generative Empathetic Data-Driven Conversational Agents for Mental Healthcare.* PhD thesis. Polytechnic University of Milan, Italy. Handle: https://hdl.handle.net/10589/202713

Shen, J., Pang, R., Weiss, R.J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R.J., Saurous, R.A., Agiomyrgiannakis, Y., Wu, Y.,

(2018). Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, Calgary, AB, Canada, 15-20 April 2018, 4779–4783.

Skerry-Ryan, R.J., Battenberg, E., Xiao, Y., Wang, Y., Stanton, D., Shor, J., Weiss, R.J., Clark, R., Saurous, R. A, (2018). Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. In Dy, J., G., Krause, A., *Proceedings of the 35th International Conference on Machine Learning, ICML 2018,* Stockholmsmässan, Stockholm, Sweden, 10-15 July 2018, vol.80 of Proceedings of Machine Learning Research, 4700–4709.

Stanton, D., Wang, Y., Skerry-Ryan, R.J., (2021). Predicting expressive speaking style from text in end-to-end speech synthesis. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018*, Athens, Greece, 18-21 December 2018, 595–602.

Tan, X, Qin, T., Soong, F.K., Liu, T.Y., (2021). *A survey on neural speech synthesis*. CoRR, abs/2106.15561.

Valle, R., Li, J., Prenger, R., Catanzaro, B., (2020). Mellotron: Multispeaker expressive voice synthesis by conditioning on rhythm, pitch and global style tokens. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020*, Barcelona, Spain, 4-8 May 4-8 2020, 6189–6193.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A.W., Kavukcuoglu, K., (2016). Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop*, Sunnyvale, CA, USA, 13-15 September 2016, 125.

Wang, Y., Skerry-Ryan, R.J., Stanton, D., Wu, Y., Weiss, R.J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q.V., Agiomyrgiannakis, Y., Saurous Clark, R., Saurous, R.A., (2017). Tacotron: Towards end-to-end speech synthesis. In Lacerda, F., *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, Stockholm, Sweden, 20-24 August 2017, 4006–4010.

Wang, Y., Stanton, D., Zhang, Y., Skerry-Ryan, R.J., Battenberg, E., Shor, J., Xiao, Y., Jia, Y., Ren, F., Saurous, R.A., (2018). Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In Dy, J.G., Krause, A., *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Stockholmsmässan, Stockholm, Sweden, 10-15 July 2018, vol. 80 of Proceedings of Machine Learning Research, 5167–5176.

Zhang, Y., Sun, S., Galley, M., Chen, Y., Brockett, C., Gao, X., Gao, J., Liu, J., Dolan, B., (2020). DIALOGPT: Large-scale generative pre-training for conversational response generation. In Celikyilmaz, A., Wen, T.-H., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020*, Online, 5-10 July 5-10 2020, 270–278.