MIXINGBOARD:
a Knowledgeable Stylized Integrated Text Generation Platform

Xiang Gao   **Michel Galley**   **Bill Dolan**

Microsoft Research, Redmond, WA, USA

{xiag,mgalley,billdol}@microsoft.com

Abstract

We present MIXINGBOARD, a platform for quickly building demos with a focus on knowledge grounded stylized text generation. We unify existing text generation algorithms in a shared codebase and further adapt earlier algorithms for constrained generation. To borrow advantages from different models, we implement strategies for cross-model integration, from the token probability level to the latent space level. An interface to external knowledge is provided via a module that retrieves on-the-fly relevant knowledge from passages on the web or any document collection. A user interface for local development, remote web-page access, and a RESTful API are provided to make it simple for users to build their own demos [1] .

1 Introduction

Neural text generation algorithms have seen great improvements over the past several years (Radford et al., 2019; Gao et al., 2019a). However each algorithm and neural model usually focuses on a specific task and may differ significantly from each other in terms of architecture, implementation, interface, and training domains. It is challenging to unify these algorithms theoretically, but a framework to organically integrate multiple algorithms and components can benefit the community in several ways, as it provides (1) a shared codebase to reproduce and compare the state-of-the-art algorithms from different groups without time consuming trial and errors, (2) a platform to experiment the cross-model integration of these algorithms, and (3) a framework to build demo quickly upon these components. This framework can be built upon existing deep learning libraries (Paszke et al., 2019;
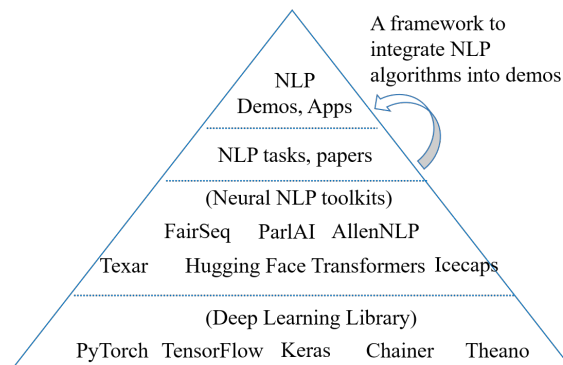

Figure 1: MIXINGBOARD is designed as a platform to organically and quickly integrate separate NLP algorithms into compelling demos

Abadi et al., 2015) and neural NLP toolkits (HuggingFace, 2019; Gardner et al., 2018; Hu et al., 2018; Ott et al., 2019; Shiv et al., 2019; Miller et al., 2017)[2], as illustrated in Fig. 1.

There are several challenges to do such integration. Firstly, engineering efforts are needed to unify the interface of different implementation. Secondly, a top-level manager needs to be designed to utilize different models together. Finally, different models are trained using different data with different performance. Cross-model integration, instead of calling each isolated model individually, can potentially improve the overall performance. In this work, we unified the models of different implementation in a single codebase, implemented demos as top-level managers to access different models, and provide strategies to allow more organic integration across the models, including token probability interpolation, cross-mode scoring, latent interpolation, and unified hypothesis ranking.

This work is also aimed to promote the development of grounded text generation. The exist-

---

[1] Source code available at github.com/microsoft/MixingBoard

[2] Although multiple libraries and toolkits are mentioned in Fig. 1, the current implementation is primarily based on PyTorch models

ing works focusing on the knowledge grounded text generation (Prabhumoye et al., 2019; Qin et al., 2019; Galley et al., 2019) usually assume the knowledge passage is given. However in practice this is not true. We provide the component to retrieve knowledge passage on-the-fly from web or customized document, to allow engineers or researchers test existing or new generation models. Keyphrase constrained generation (Hokamp and Liu, 2017) is another type of grounded generation, broadly speaking. Similarly the keyphrase needs to be provided to apply such constraints. We provide tools to extract constraints from knowledge passage or stylized corpus.

Finally, friendly user interface is a component usually lacking in the implementation of neural models but it is necessary for a demo-centric framework. We provide scripts to build local terminal demo, webpage demo, and RESTful API demo.

2 Design

Our goal is to build a framework that will allow users to quickly build text generation demos using existing modeling techniques. This design allows the framework to be almost agnostic to the ongoing development of text generation techniques (Gao et al., 2019a). Instead, we focus on the organic integration of models and the interfaces for the final demo/app.

From a top-down view, our design are bounded to two markets: text processing assistant, and conversational AI, as illustrated in Fig. 2. Two demos are present as examples in these domains: document auto-completion and Sherlock Holmes. We further breakdown these demos into several tasks, designed to be shared across different demos. We also designed several strategies to integrate multiple models to generate text. These strategies allow each model to plug-in without heavy constraints on the architecture of the models, as detailed in Section 4.

As the goal is not another deep learning NLP toolkit, we rely on existing ones (HuggingFace, 2019; Paszke et al., 2019; Gardner et al., 2018) and online API services Bing Web Search provided in Azure Cognitive Service[3] and TagME.[4] Similarly, most tasks are using existing algorithms: language modeling (Zhang et al., 2019; Radford et al.,

2019), knowledge grounded generation (Qin et al., 2019; Prabhumoye et al., 2019) or span retrieval (Seo et al., 2016; Devlin et al., 2018), style transfer (Gao et al., 2019c,b) and constrained generation (Hokamp and Liu, 2017).

3 Modules

3.1 Knowledge passage retrieval

We use the following free-text, unstructured text sources to retrieve relevant knowledge passage.

- Search engine. Free-text form "knowledge" is retrieved from the following sources 1) text snippets from the (customized) webpage search; 2) text snippets (customized) news search; 3) user-provided documents.
- Specialized websites. For certain preferred domains, e.g., wikipedia.org, we will further download the whole webpage (rather than just the text snippet returned from search engine) to obtain more text snippets.
- Users can also provide their customized knowledge base, like a README file, which can be updated on-the-fly, to allow the agent using latest knowledge.

User may select one or multiple sources listed above to obtain knowledge passage candidates. Then the text snippets are then ranked by relevancy as well as diversity.

3.2 Stylized synonym

We provide a component to retrieve synonym of given target style for a query word. This component is useful for the style transfer module (Section 3.4) as well as the constrained generation module (Section 3.7).

The similarity based on word2vec, $\text{sim}_{\text{word2vec}}$, is defined as the cosine similarity between the vectors of two words. The similarity based on human-edited dictionary, sim_{dict}, is defined as 1 if the candidate word in the synonym list of the query word, otherwise 0. The final similarity between the two words is defined as the weighted average of these two similarities:

$$\text{sim} = (1 - w_{\text{dict}}) \, \text{sim}_{\text{word2vec}} + w_{\text{dict}} \, \text{sim}_{\text{dict}}$$

We only choose the candidate word with a similarity higher than certain threshold as the synonym of the query word. Then we calculate the style score of these synonym using a style classifier. We
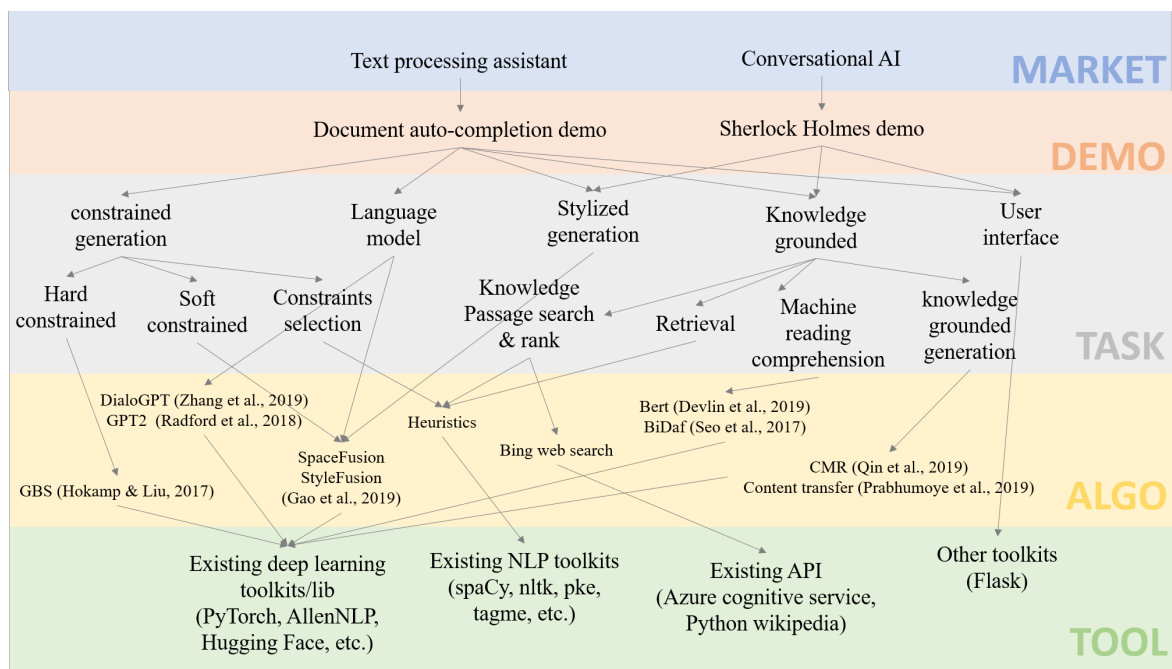
Figure 2: The architecture of MIXINGBOARD, consisting of layers from basic tools, algorithms, tasks to integrated demos with market into consideration.

provided a logistic regression model taking 1gram multi-hot vector as features, trained on non-stylized corpus vs. stylized corpus.

3.3 Latent interpolation

For a given two latent vectors, z_a and z_b, we expect the decoded results from the interpolated vector $z_i = u z_a + (1 - u) z_b$ can retain the desired features from both z_a and z_b. However this requires a interpolatable, smooth latent space. For this purpose, we apply the SpaceFusion (Gao et al., 2019b) and StyleFusion (Gao et al., 2019c) to learn such latent space. The latent interpolation is then used to transfer style, apply soft constraints, and interpolating hypothesis obtained using different models.

3.4 Stylized generation

Gao et al. (2019c) proposed StyleFusion to generate stylized response for a given conversation context by structuring a shared latent space for non-stylized conversation data and stylized samples. We extend it to a style transfer method, i.e., modify the style of a input sentence while maintaining its content, via latent interpolation (see Section 3.3).

- **Soft-edit** refers to a two-step algorithm, 1) edit the input sentence by replace each word by a synonym of the target style (e.g. "cookie" replaced by "biscuit" if the target style is

British), if there exists any; 2) the edited sentence from step 1 may not be fluent, so we then apply latent interpolation between the input sentence and edited sentence to seek a sentence that is both stylized and fluent.

- **Soft-retrieval** refers to a similar two-step algorithm, but step 1) is to retrieve a "similar" sentence from a stylized corpus, and then apply step 2) to do the interpolation. One example is given in Fig. 5. The hypothesis "he was once a schoolmaster in the north of england" is retrieved given the DialoGPT hypothesis "he's a professor at the university of london".

3.5 Conditioned text generation

Generate a set of candidate responses conditioned on the conversation history, or a follow-up sentence conditioned on the existing text.

- GPT-2 (Radford et al., 2019) is a transformer (Vaswani et al., 2017) based text generation model.
- DialoGPT (Zhang et al., 2019) is a large-scale pre-trained conversation model obtained by training GPT-2 (Radford et al., 2019) on Reddit comments data.
- SpaceFusion (Gao et al., 2019b) is a regularized multi-task learning framework proposed to learn a smooth and interpolatable latent space.

226

3.6 Knowledge grounded generation

We consider the following methods to consume the retrieved knowledge passage and relevant long-form text on the fly as a source of external knowledge.

- Machine reading comprehension. In the codebase, we fine-tuned BERT on SQuAD.
- Content transfer is a task proposed in (Prabhumoye et al., 2019) designed to, given a context, generate a sentence using knowledge from an external article. We implemented this algorithm in the codebase.
- Knowledge grounded response generation is a task firstly proposed in (Ghazvininejad et al., 2018) and later extended in Dialog System Technology Challenge 7 (DSTC7)(Galley et al., 2019). We implemented the CMR algorithm (Conversation with on-demand Machine Reading) proposed in (Qin et al., 2019).

3.7 Constrained generation

Besides the grounded generation, it is also useful to apply constraints at the decoding stage that encourage the generated hypotheses contain the desired phrases. We provide the following two ways to obtain constraints.

- Key phrases extracted from the Knowledge passage. We use the PKE package (Boudin, 2016) to identify the keywords.
- In some cases, users may want to use a stylized version of the topic phrases or phrase of a desired style as the constraints. We use the stylized synonym algorithm as introduced in Section 3.4 to provide such stylized constraints.

With the constraints obtained above, we provide the following two ways to apply such constraints during decoding.

- **Hard constraint** is applied via Grid Beam Search (GBS) (Hokamp and Liu, 2017), which is a lexically constrained decoding algorithm that can be applied to almost any models at the decoding stage and generate hypotheses that contain desired paraphrases (i.e. the constraints). We implemented GBS to provide a hard constrained decoding.
- **Soft constraint** refers the case that generation is likely, but not always, to satisfy constraints (e.g. include given keywords in the
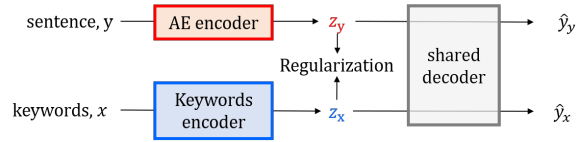

Figure 3: A soft keywords constrained generation model based on SpaceFusion (Gao et al., 2019b).

hypothesis). We provide an adapted version of SpaceFusion (Gao et al., 2019b) for this purpose. Gao et al. (2019b) proposed to align the latent space of a Sequence-to-Sequence (S2S) model and that of an Autoencoder (AE) model to improve dialogue generation performance. Inspired by this work, we proposed to replace the S2S model by a keywords-to-sequence model, which takes multi-hot input of the keywords x identified from sentence y, as illustrated in Fig. 3. During training, we simply choose the top-k rare words (rareness measured by inverse document frequency) as the keywords, and k is randomly choose from a Uniform distribution $k \sim U(1, K)$.

4 Cross-model integration

Multiple models may be called for the same query and returns different responses. We propose the following ways to organically integrate multiple models, as illustrated in Fig. 4. User can apply these strategies with customized models.

- **Token probability interpolation** refers prediction of the next token using a (weighted) average of the token probability distributions from two or more models given the same time step and given the same context and incomplete hypothesis. Previously, it has been proposed to bridge a conversation model and stylized language model (Niu and Bansal, 2018). This technique does not require the models to share the latent space but the vocabulary should be shared across different models.
- **Latent interpolation** refers the technique introduced in Section 3.3. It provides a way to interpolate texts in the latent space. Unlike the token-level strategy introduced above, this technique focuses on the latent level and ingests information from the whole sentence. However if the two candidates are too dissimilar, the interpolation may result in undesired outputs. The soft constraint algorithm intro-
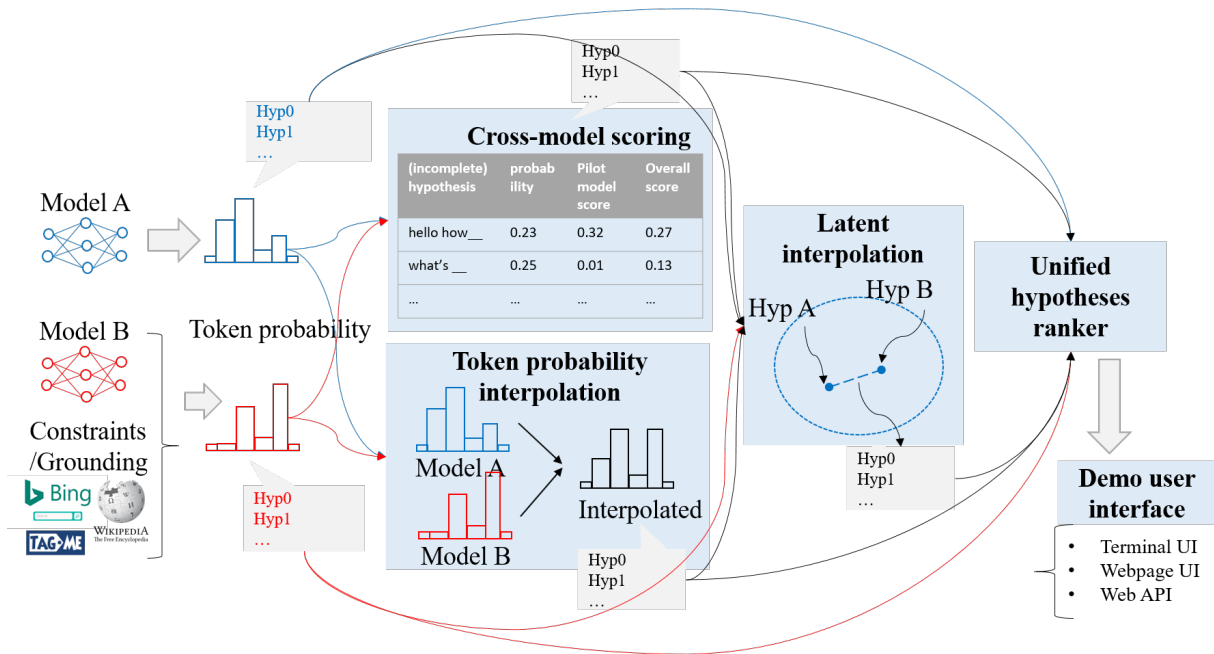
Figure 4: An example flow chart showing the integration of two models at different stages (blue boxes).

duced in Section 3.7 is one option to apply such interpolation.

- **Cross model pruning** refers pruning the hypothesis candidates (can be incomplete hypothesis, e.g. during beam search) not just based on the joint token probability, but also the evaluated probability from a secondary model. This strategy does not require a shared vocabulary or a shared latent space. Interpolating two models trained on dissimilar domains may be risky but the cross model pruning strategy is safer as the secondary model is only used roughly as a discriminator rather than a generator.

- **Unified hypothesis ranking** is the final step which sum up the hypotheses generated from each single model and these from the integration of multiple models using the above strategies. We consider the following criteria for the hypothesis ranking: 1) likelihood, measured by the conditional token probability given the context; 2) informativeness, measured by average inverse document frequency (IDF) of the tokens in the hypothesis; 3) repetition penalty, measured by the ratio of the number of unique ngrams and the number of total ngrams. and 4) style intensity, measured by a style classifiers, if style is considered.

5 Demos

5.1 Virtual Sherlock Holmes

This demo is a step towards a virtual version of Sherlock Holmes, able to chat in Sherlock Holmes style, with Sherlock Holmes background knowledge in mind. As an extended version of the one introduced by Gao et al. (2019c), the current demo is grounded on knowledge and coupled with more advanced language modeling (Zhang et al., 2019). It is designed to integrate the following components: open-domain conversation, stylized response generation, knowledge-grounded conversation, and question answering. Specifically, for a given query, the following steps are executed:

- Call DialoGPT (Zhang et al., 2019) and Style-Fusion (Gao et al., 2019c) to get a set of hypotheses.
- Call the knowledge passage selection module to get a set of candidate passages. Then feed these passages to the span selection algorithm (Bert-based MRC (Devlin et al., 2018)) and CMR (Qin et al., 2019) to get a set of knowledge grounded response.
- Optionally, use the cross-model integration strategies, such as interpolating the token probability of DialoGPT and CMR.
- Based on TF-IDF similarity, best answer is retrieved from a user provided corpus of question-answer pairs. If the similarity is

lower than certain threshold, the retrieved result will not be returned.

- Apply the style transfer module to obtain stylized version of the the hypotheses obtained from steps above.
- feed all hypotheses to the unified ranker and return the top ones.

5.2 Document auto-completion assistant

This demo is designed as a writing assistant, which provides suggestion of the next sentence given the context. The assistant is expected to be knowledgeable (able to retrieve relevant knowledge passage from web or a given unstructured text source) and stylized (if a given target style is specified). For a given query, the following steps are executed:

- Call language model GPT2 (Radford et al., 2019) to get a set of hypotheses
- Call the knowledge passage selection module to get a set of candidate passages. Then feed these passages to content transfer algorithm (Prabhumoye et al., 2019) to get a set of knowledge grounded response.
- Optionally, use the cross-model integration strategies, such as latent interpolation to interpolate hypotheses from above models.
- Apply the style transfer module to obtain stylized version of the the hypotheses obtained from steps above.
- Feed all hypotheses to the unified ranker and return the top ones.

6 User interface

We provided the following three ways to access the demos introduced above for local developer, remote human user, and interface for other programs.

- **Command line interface** is provided for local interaction. This is designed for developer to test the codebase.
- **Webpage interface** is implemented using the Flask toolkit.[5] A graphic interface is provided with html webpage for remote access for human user. As illustrated in Fig. 5, the Sherlock Holmes webpage consists of a input panel where the user can provide context and control style, a hypothesis list which specify the model and scores of the ranked hypotheses, and a knowledge passage list showing
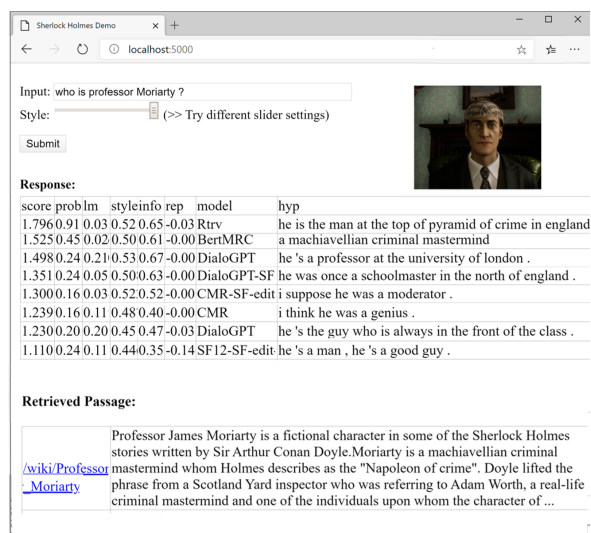

Figure 5: Sherlock Holmes webpage demo with wikipedia knowledge example.

the retrieved knowledge passages. Another example is given in Fig. 6 for document auto-completion demo, where multiple options of knowledge passage is given.

- **RESTful API** is implemented using the Flask-RESTful toolkit.[6] JSON object will be returned for remote request. This interface is designed to allow remote access for other programs. One example is to host this RESTful API on a dedicated GPU machine, so the webpage interface can be hosted on another less powerful machine to send request through RESTful API.

7 Conclusion

MIXINGBOARD is a new open-source platform to organically integrate multiple state-of-the-art NLP algorithms to build demo quickly with user friendly interface. We unified these NLP algorithms in a single codebase, implemented demos as top-level managers to access different models, and provide strategies to allow more organic integration across the models. We provide the component to retrieve knowledge passage on-the-fly from web or customized document for grounded text generation. For future work, we plan to keep adding the state-of-the-art algorithms, reduce latency and fine-tune the implemented models on larger and/or more comprehensive corpus to improve performance.
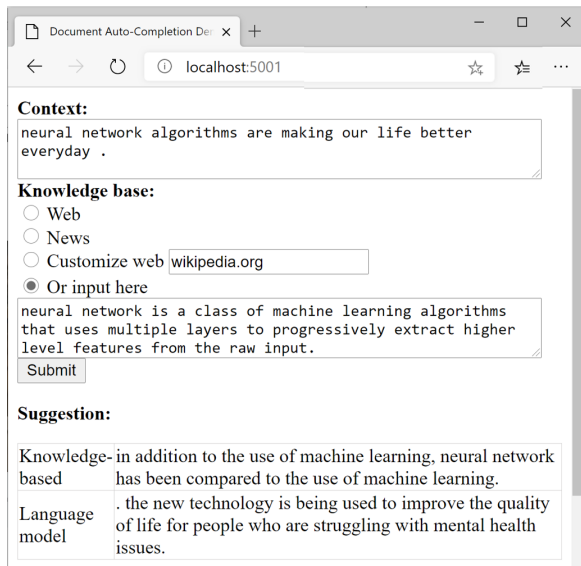
---

[5] https://flask.palletsprojects.com/en/1.1.x/

[6] https://flask-restful.readthedocs.io/en/latest/

Figure 6: Document Auto-completion webpage demo with user input knowledge passage.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Florian Boudin. 2016. pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 69–73, Osaka, Japan.

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

Michel Galley, Chris Brockett, Xiang Gao, Jianfeng Gao, and Bill Dolan. 2019. Grounded response generation task at DSTC7. In *AAAI Dialog System Technology Challenges (DSTC7) Workshop*.

Jianfeng Gao, Michel Galley, and Lihong Li. 2019a. Neural approaches to conversational AI. *Foundations and Trends in Information Retrieval*, 13(2-3):127–298.

Xiang Gao, Sungjin Lee, Yizhe Zhang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2019b.

Jointly optimizing diversity and relevance in neural response generation. *NAACL-HLT 2019*.

Xiang Gao, Yizhe Zhang, Sungjin Lee, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2019c. Structuring latent spaces for stylized response generation. *arXiv preprint arXiv:1909.05361*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. *arXiv preprint arXiv:1704.07138*.

Zhiting Hu, Haoran Shi, Zichao Yang, Bowen Tan, Tiancheng Zhao, Junxian He, Wentao Wang, Lianhui Qin, Di Wang, et al. 2018. Texar: A modularized, versatile, and extensible toolkit for text generation. *arXiv preprint arXiv:1809.00794*.

HuggingFace. 2019. PyTorch transformer repository. https://github.com/huggingface/pytorch-transformers.

Alexander H Miller, Will Feng, Adam Fisch, Jiasen Lu, Dhruv Batra, Antoine Bordes, Devi Parikh, and Jason Weston. 2017. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *Transactions of the Association of Computational Linguistics*, 6:373–389.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Shrimai Prabhumoye, Chris Quirk, and Michel Galley. 2019. Towards content transfer through grounded text generation. In *Proc. of NAACL*.

Lianhui Qin, Michel Galley, Chris Brockett, Xiaodong Liu, Xiang Gao, Bill Dolan, Yejin Choi, and Jianfeng Gao. 2019. Conversing by reading: Contentful neural conversation with on-demand machine reading. In *Proc. of ACL*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Vighnesh Leonardo Shiv, Chris Quirk, Anshuman Suri, Xiang Gao, Khuram Shahid, Nithya Govindarajan, Yizhe Zhang, Jianfeng Gao, Michel Galley, Chris Brockett, et al. 2019. Microsoft icecaps: An open-source toolkit for conversation modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics: System Demonstrations*, pages 123–128.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. DialoGPT: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.