Generating Personalized Recipes from Historical User Preferences

Bodhisattwa Prasad Majumder*, **Shuyang Li***, **Jianmo Ni, Julian McAuley**
Computer Science and Engineering
University of California, San Diego
{bmajumde, shl008, jin018, jmcauley}@ucsd.edu

Abstract

Existing approaches to recipe generation are unable to create recipes for users with culinary preferences but incomplete knowledge of ingredients in specific dishes. We propose a new task of *personalized recipe generation* to help these users: expanding a name and incomplete ingredient details into complete natural-text instructions aligned with the user's historical preferences. We attend on technique- and recipe-level representations of a user's previously consumed recipes, fusing these 'user-aware' representations in an attention fusion layer to control recipe text generation. Experiments on a new dataset of 180K recipes and 700K interactions show our model's ability to generate plausible and personalized recipes compared to non-personalized baselines.

1 Introduction

In the kitchen, we increasingly rely on instructions from cooking websites: recipes. A cook with a predilection for Asian cuisine may wish to prepare chicken curry, but may not know all necessary ingredients apart from a few basics. These users with limited knowledge cannot rely on existing recipe generation approaches that focus on creating coherent recipes given all ingredients and a recipe name (Kiddon et al., 2016). Such models do not address issues of personal preference (e.g. culinary tastes, garnish choices) and incomplete recipe details. We propose to approach both problems via *personalized generation* of plausible, user-specific recipes using user preferences extracted from previously consumed recipes.

Our work combines two important tasks from natural language processing and recommender systems: data-to-text generation (Gatt and Krahmer, 2018) and personalized recommendation (Rashid et al., 2002). Our model takes as user input the name of a specific dish, a few key ingredients, and a calorie level. We pass these loose input specifications to an encoder-decoder framework and attend on user profiles—learned latent representations of recipes previously consumed by a user—to generate a recipe *personalized* to the user's tastes. We fuse these 'user-aware' representations with decoder output in an attention fusion layer to jointly determine text generation. Quantitative (perplexity, user-ranking) and qualitative analysis on user-aware model outputs confirm that personalization indeed assists in generating plausible recipes from incomplete ingredients.

While personalized text generation has seen success in conveying user writing styles in the product review (Ni et al., 2017; Ni and McAuley, 2018) and dialogue (Zhang et al., 2018) spaces, we are the first to consider it for the problem of recipe generation, where output quality is heavily dependent on the *content* of the instructions—such as ingredients and cooking techniques.

To summarize, our main contributions are as follows:

1. We explore a new task of generating plausible and personalized recipes from incomplete input specifications by leveraging historical user preferences;[1]
2. We release a new dataset of 180K+ recipes and 700K+ user reviews for this task;
3. We introduce new evaluation strategies for generation quality in instructional texts, centering on quantitative measures of coherence. We also show qualitatively and quantitatively that personalized models generate high-quality and specific recipes that align with historical user preferences.

* denotes equal contribution

[1] Our source code and appendix are at https://github.com/majumderb/recipe-personalization

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 5976–5982,
Hong Kong, China, November 3–7, 2019.

2 Related Work

Large-scale transformer-based language models have shown surprising expressivity and fluency in creative and conditional long-text generation (Vaswani et al., 2017; Radford et al., 2019). Recent works have proposed hierarchical methods that condition on narrative frameworks to generate internally consistent long texts (Fan et al., 2018; Xu et al., 2018; Yao et al., 2018). Here, we generate procedurally structured recipes instead of free-form narratives.

Recipe generation belongs to the field of data-to-text natural language generation (Gatt and Krahmer, 2018), which sees other applications in automated journalism (Leppänen et al., 2017), question-answering (Agrawal et al., 2017), and abstractive summarization (Paulus et al., 2018), among others. Kiddon et al. (2015); Bosselut et al. (2018b) model recipes as a structured collection of ingredient entities acted upon by cooking actions. Kiddon et al. (2016) imposes a 'checklist' attention constraint emphasizing hitherto unused ingredients during generation. Yang et al. (2017) attend over explicit ingredient references in the prior recipe step. Similar hierarchical approaches that infer a full ingredient list to constrain generation will not help personalize recipes, and would be infeasible in our setting due to the potentially unconstrained number of ingredients (from a space of 10K+) in a recipe. We instead learn historical preferences to guide full recipe generation.

A recent line of work has explored user- and item-dependent aspect-aware review generation (Ni et al., 2017; Ni and McAuley, 2018). This work is related to ours in that it combines contextual language generation with personalization. Here, we attend over historical user preferences from previously consumed recipes to generate recipe content, rather than writing styles.

3 Approach

Our model's input specification consists of: the recipe name as a sequence of tokens, a partial list of ingredients, and a caloric level (high, medium, low). It outputs the recipe instructions as a token sequence: $\mathcal{W}_r = \{w_{r,0}, \ldots, w_{r,T}\}$ for a recipe r of length T. To personalize output, we use historical recipe interactions of a user $u \in \mathcal{U}$.

Encoder: Our encoder has three embedding layers: vocabulary embedding $\mathcal{V}$, ingredient embedding $\mathcal{I}$, and caloric-level embedding $\mathcal{C}$. Each token in the (length L_n) recipe name is embedded via $\mathcal{V}$; the embedded token sequence is passed to a two-layered bidirectional GRU (BiGRU) (Cho et al., 2014), which outputs hidden states for names $\{\mathbf{n}_{\text{enc},j} \in \mathbb{R}^{2d_h}\}$, with hidden size d_h. Similarly each of the L_i input ingredients is embedded via $\mathcal{I}$, and the embedded ingredient sequence is passed to another two-layered BiGRU to output ingredient hidden states as $\{\mathbf{i}_{\text{enc},j} \in \mathbb{R}^{2d_h}\}$. The caloric level is embedded via $\mathcal{C}$ and passed through a projection layer with weights W_c to generate calorie hidden representation $\mathbf{c}_{\text{enc}} \in \mathbb{R}^{2d_h}$.

Ingredient Attention: We apply attention (Bahdanau et al., 2015) over the encoded ingredients to use encoder outputs at each decoding time step. We define an attention-score function α with key K and query Q:

$$\alpha(K, Q) = \frac{\exp\left(\tanh\left(W_\alpha\left[K + Q\right] + \mathbf{b}_\alpha\right)\right)}{Z},$$

with trainable weights W_α, bias $\mathbf{b}_\alpha$, and normalization term Z. At decoding time t, we calculate the ingredient context $\mathbf{a}_t^i \in \mathbb{R}^{d_h}$ as:

$$\mathbf{a}_t^i = \sum_{j=1}^{L_i} \alpha\left(\mathbf{i}_{\text{enc},j}, \mathbf{h}_t\right) \times \mathbf{i}_{\text{enc},j}.$$

Decoder: The decoder is a two-layer GRU with hidden state h_t conditioned on previous hidden state h_{t-1} and input token $w_{r,t}$ from the original recipe text. We project the concatenated encoder outputs as the initial decoder hidden state:

$$\mathbf{h}_0 \left(\in \mathbb{R}^{d_h}\right) = W_{h_0}\left[\mathbf{n}_{\text{enc},L_n}; \mathbf{i}_{\text{enc},L_i}; \mathbf{c}_{\text{enc}}\right] + \mathbf{b}_{h_0}$$
$$\mathbf{h}_t, \mathbf{o}_t = \text{GRU}\left(\left[w_{r,t}; \mathbf{a}_t^i\right], \mathbf{h}_{t-1}\right).$$

To bias generation toward user preferences, we attend over a user's previously reviewed recipes to jointly determine the final output token distribution. We consider two different schemes to model preferences from user histories: (1) recipe interactions, and (2) techniques seen therein (defined in Section 4). Rendle et al. (2009); Quadrana et al. (2018); Ueda et al. (2011) explore similar schemes for personalized recommendation.

Prior Recipe Attention: We obtain the set of prior recipes for a user u: R_u^+, where each recipe can be represented by an embedding from a recipe embedding layer $\mathcal{R}$ or an average of the name tokens embedded by $\mathcal{V}$. We attend over the k-most recent prior recipes, R_u^{k+}, to account for temporal drift of user preferences (Moore et al., 2013).

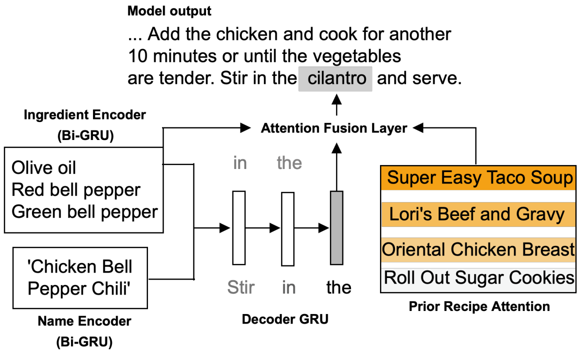

Figure 1: Sample data flow through model architecture. Emphasis on prior recipe attention scores (darker is stronger). Ingredient attention omitted for clarity.

These embeddings are used in the '**Prior Recipe**' and '**Prior Name**' models, respectively.

Given a recipe representation $\mathbf{r} \in \mathbb{R}^{d_r}$ (where d_r is recipe- or vocabulary-embedding size depending on the recipe representation) the *prior recipe attention* context $\mathbf{a}_t^{r_u}$ is calculated as

$$\mathbf{a}_t^{r_u} = \sum_{r \in R_u^{k+}} \alpha\left(\mathbf{r}, \mathbf{h}_t\right) \times \mathbf{r}.$$

Prior Technique Attention: We calculate prior technique preference (used in the '**Prior Tech**' model) by normalizing co-occurrence between users and techniques seen in R_u^+, to obtain a preference vector $\boldsymbol{\rho}_u$. Each technique x is embedded via a technique embedding layer $\mathcal{X}$ to $\mathbf{x} \in \mathbb{R}^{d_x}$. *Prior technique attention* is calculated as

$$\mathbf{a}_t^{x_u} = \sum_{x \text{ seen in } R_u^+} \left(\alpha\left(\mathbf{x}, \mathbf{h}_t\right) + \rho_{u,x}\right) \times \mathbf{x},$$

where, inspired by copy mechanisms (See et al., 2017; Gu et al., 2016), we add $\rho_{u,x}$ for technique x to emphasize the attention by the user's prior technique preference.

Attention Fusion Layer: We fuse all contexts calculated at time t, concatenating them with decoder GRU output and previous token embedding:

$$\mathbf{a}_t^f = \text{ReLU}\left(W_f\left[w_{r,t}; \mathbf{o}_t; \mathbf{a}_t^i; (\mathbf{a}_t^{r_u} \text{ or } \mathbf{a}_t^{x_u})\right] + \mathbf{b}_f\right).$$

We then calculate the token probability:

$$P(S_{r,t}) = \text{softmax}\left(W_P[\mathbf{a}_t^f] + \mathbf{b}_P\right),$$

and maximize the log-likelihood of the generated sequence conditioned on input specifications and user preferences. Figure 1 shows a case where the Prior Name model attends strongly on previously consumed savory recipes to suggest the usage of an additional ingredient ('cilantro').

Split	# Users	# Recipes	# Actions	Sparsity[3]
Train	25,076	160,901	698,901	99.983%
Dev	7,023	6,621	7,023	–
Test	12,455	11,695	12,455	–

Table 1: Statistics of Food.com interactions

4 Recipe Dataset: Food.com

We collect a novel dataset of 230K+ recipe texts and 1M+ user interactions (reviews) over 18 years (2000-2018) from Food.com.[2] Here, we restrict to recipes with at least 3 steps, and at least 4 and no more than 20 ingredients. We discard users with fewer than 4 reviews, giving 180K+ recipes and 700K+ reviews, with splits as in Table 1.

Our model must learn to generate from a diverse recipe space: in our training data, the average recipe length is 117 tokens with a maximum of 256. There are 13K unique ingredients across all recipes. Rare words dominate the vocabulary: 95% of words appear <100 times, accounting for only 1.65% of all word usage. As such, we perform Byte-Pair Encoding (BPE) tokenization (Sennrich et al., 2016; Radford et al., 2018), giving a training vocabulary of 15K tokens across 19M total mentions. User profiles are similarly diverse: 50% of users have consumed ≤6 recipes, while 10% of users have consumed >45 recipes.

We order reviews by timestamp, keeping the most recent review for each user as the test set, the second most recent for validation, and the remainder for training (sequential leave-one-out evaluation (Kang and McAuley, 2018)). We evaluate only on recipes not in the training set.

We manually construct a list of 58 cooking techniques from 384 cooking actions collected by Bosselut et al. (2018b); the most common techniques (*bake*, *combine*, *pour*, *boil*) account for 36.5% of technique mentions. We approximate technique adherence via string match between the recipe text and technique list.

5 Experiments and Results

For training and evaluation, we provide our model with the first 3-5 ingredients listed in each recipe. We decode recipe text via top-k sampling (Radford et al., 2019), finding $k = 3$ to produce satisfactory results. We use a hidden size $d_h = 256$

[2]https://www.kaggle.com/shuyangli94/food-com-recipes-and-user-interactions

[3]Ratio of unobserved actions to all possible actions.

Model	BPE PPL	BLEU-1	BLEU-4	ROUGE-L	D-1 (%)	D-2 (%)	UMA	MRR	PP (%)
NN	–	20.279	0.465	16.871	0.931	9.394	0.100	0.293	–
Enc-Dec	9.611	28.391	**3.385**	**25.001**	0.220	1.928	0.100	0.293	–
Prior Tech	9.572	**28.864**	3.312	24.920	0.233	**2.158**	0.128	0.319	62.821
Prior Recipe	9.551	27.858	3.215	24.822	0.231	2.062	0.302	0.412	**66.026**
Prior Name	**9.516**	28.046	3.211	24.794	**0.233**	2.080	**0.505**	**0.628**	61.165

Table 2: Metrics on generated recipes from test set. D-1/2 = Distinct-1/2, UMA = User Matching Accuracy, MRR = Mean Reciprocal Rank, PP = Pairwise preference over baseline (evaluated for 310 recipe pairs per model).

Input	**Name:** Pomberrytini; **Ingredients:** pomegranate-blueberry juice, cranberry juice, vodka; **Calorie:** Low
Gold	Place everything except the orange slices in a cocktail shaker. Shake until well mixed and well chilled. Pour into martini glasses and float an orange slice in each glass.
Enc-Dec	Combine all ingredients. Cover and refrigerate. Serve with whipped topping.
Prior Tech	Combine all ingredients. Store in refrigerator. Serve over ice. Enjoy!
Prior Recipe	Pour the ice into a cocktail shaker. Pour in the vodka and vodka. Add a little water and shake to mix. Pour into the glass and garnish with a slice of orange slices. Enjoy!
Prior Name	Combine all ingredients except for the ice in a blender or food processor. Process to make a smooth paste and then add the remaining vodka and blend until smooth. Pour into a chilled glass and garnish with a little lemon and fresh mint.

Table 3: Sample generated recipe. Emphasis on personalization and explicit ingredient mentions via highlights.

for both the encoder and decoder. Embedding dimensions for vocabulary, ingredient, recipe, techniques, and caloric level are 300, 10, 50, 50, and 5 (respectively). For prior recipe attention, we set $k = 20$, the 80th %-ile for the number of user interactions. We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 10^{-3}, annealed with a decay rate of 0.9 (Howard and Ruder, 2018). We also use teacher-forcing (Williams and Zipser, 1989) in all training epochs.

In this work, we investigate how leveraging historical user preferences can improve generation quality over strong baselines in our setting. We compare our personalized models against two baselines. The first is a name-based Nearest-Neighbor model (**NN**). We initially adapted the Neural Checklist Model of Kiddon et al. (2016) as a baseline; however, we ultimately use a simple Encoder-Decoder baseline with ingredient attention (**Enc-Dec**), which provides comparable performance and lower complexity. All personalized models outperform baseline in BPE perplexity (Table 2) with Prior Name performing the best. While our models exhibit comparable performance to baseline in BLEU-1/4 and ROUGE-L, we generate more diverse (Distinct-1/2: percentage of distinct unigrams and bigrams) and acceptable recipes. BLEU and ROUGE are not the most appropriate metrics for generation quality. A 'correct' recipe can be written in many ways with the same main entities (ingredients). As BLEU-1/4 capture structural information via n-gram matching, they are not correlated with subjective recipe quality. This mirrors observations from Baheti et al. (2018); Fan et al. (2018).

We observe that personalized models make more diverse recipes than baseline. They thus perform better in BLEU-1 with more key entities (ingredient mentions) present, but worse in BLEU-4, as these recipes are written in a personalized way and deviate from gold on the phrasal level. Similarly, the 'Prior Name' model generates more unigram-diverse recipes than other personalized models and obtains a correspondingly lower BLEU-1 score.

Qualitative Analysis: We present sample outputs for a cocktail recipe in Table 3, and additional recipes in the appendix. Generation quality progressively improves from generic baseline output to a blended cocktail produced by our best performing model. Models attending over prior recipes explicitly reference ingredients. The Prior Name model further suggests the addition of lemon and mint, which are reasonably associated with previously consumed recipes like coconut mousse and pork skewers.

Personalization: To measure personalization, we evaluate how closely the generated text corresponds to a particular user profile. We compute the likelihood of generated recipes using identical input specifications but conditioned on ten different user profiles—one 'gold' user who consumed the original recipe, and nine randomly generated user profiles. Following Fan et al. (2018), we expect the highest likelihood for the recipe conditioned on the gold user. We measure user matching accuracy (UMA)—the proportion where the gold user is ranked highest—and Mean Reciprocal Rank (MRR) (Radev et al., 2002) of the gold user. All personalized models beat baselines in both metrics, showing our models personalize generated recipes to the given user profiles. The Prior Name model achieves the best UMA and MRR by a large margin, revealing that prior recipe names are strong signals for personalization. Moreover, the addition of attention mechanisms to capture these signals improves language modeling performance over a strong non-personalized baseline.

Recipe Level Coherence: A plausible recipe should possess a coherent step order, and we evaluate this via a metric for recipe-level coherence. We use the neural scoring model from Bosselut et al. (2018a) to measure recipe-level coherence for each generated recipe. Each recipe step is encoded by BERT (Devlin et al., 2019). Our scoring model is a GRU network that learns the overall recipe step ordering structure by minimizing the cosine similarity of recipe step hidden representations presented in the correct and reverse orders. Once pretrained, our scorer calculates the similarity of a generated recipe to the forward and backwards ordering of its corresponding gold label, giving a score equal to the difference between the former and latter. A higher score indicates better step ordering (with a maximum score of 2). Table 4 shows that our personalized models achieve average recipe-level coherence scores of 1.78-1.82, surpassing the baseline at 1.77.

Recipe Step Entailment: Local coherence is also crucial to a user following a recipe: it is crucial that subsequent steps are logically consistent with prior ones. We model local coherence as an entailment task: predicting the likelihood that a recipe step follows the preceding. We sample several consecutive (positive) and non-consecutive (negative) pairs of steps from each recipe. We train a BERT (Devlin et al., 2019) model to predict the entailment score of a pair of steps separated by a `[SEP]` token, using the final representation of the `[CLS]` token. The step entailment score is computed as the average of scores for each set of consecutive steps in each recipe, averaged over every generated recipe for a model, as shown in Table 4.

Model	Recipe Level Coherence	Recipe Step Entailment
Enc-Dec	1.77	0.72
Prior Tech	1.78	0.73
Prior Recipe	1.80	0.76
Prior Name	**1.82**	**0.78**

Table 4: Coherence metrics on generated recipes from test set.

Human Evaluation: We presented 310 pairs of recipes for pairwise comparison (Fan et al., 2018) (details in appendix) between baseline and each personalized model, with results shown in Table 2. On average, human evaluators preferred personalized model outputs to baseline 63% of the time, confirming that personalized attention improves the semantic plausibility of generated recipes. We also performed a small-scale human coherence survey over 90 recipes, in which 60% of users found recipes generated by personalized models to be more coherent and preferable to those generated by baseline models.

6 Conclusion

In this paper, we propose a novel task: to generate personalized recipes from incomplete input specifications and user histories. On a large novel dataset of 180K recipes and 700K reviews, we show that our personalized generative models can generate plausible, personalized, and coherent recipes preferred by human evaluators for consumption. We also introduce a set of automatic coherence measures for instructional texts as well as personalization metrics to support our claims. Our future work includes generating structured representations of recipes to handle ingredient properties, as well as accounting for references to collections of ingredients (e.g. "dry mix").

Acknowledgements. This work is partly supported by NSF #1750063. We thank all reviewers for their constructive suggestions, as well as Rei M., Sujoy P., Alicia L., Eric H., Tim S., Kathy C., Allen C., and Micah I. for their feedback.

References

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. 2017. VQA: visual question answering. *IJCV*, 123(1):4–31.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. In *EMNLP*.

Antoine Bosselut, Asli Çelikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. 2018a. Discourse-aware neural rewards for coherent text generation. In *NAACL-HLT*.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018b. Simulating action dynamics with neural process networks. In *ICLR*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.*, 61:65–170.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. 2015. Mise en place: Unsupervised interpretation of instructional recipes. In *EMNLP*.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Leo Leppänen, Myriam Munezero, Mark Granroth-Wilding, and Hannu Toivonen. 2017. Data-driven news generation for automated journalism. In *INLG*.

Joshua L. Moore, Shuo Chen, Douglas Turnbull, and Thorsten Joachims. 2013. Taste over time: The temporal dynamics of user preferences. In *ISMIR*.

Jianmo Ni, Zachary C. Lipton, Sharad Vikram, and Julian McAuley. 2017. Estimating reactions and recommending products with generative models of reviews. In *IJCNLP*.

Jianmo Ni and Julian McAuley. 2018. Personalized review generation by expanding phrases and attending on aspect-aware representations. In *ACL*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *ICLR*.

Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-aware recommender systems. In *UMAP*.

Dragomir R. Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating web-based question answering systems. In *LREC*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. 2002. Getting to know you: learning new user preferences in recommender systems. In *IUI*.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: bayesian personalized ranking from implicit feedback. In *UAI*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima. 2011. User's food preference extraction for personalized cooking recipe recommendation. In *SPIM*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *EMNLP*.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In *EMNLP*.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2018. Plan-and-write: Towards better automatic storytelling. *CoRR*, abs/1811.05701.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *ACL*.