Counterfactual Language Model Adaptation for Suggesting Phrases

Kenneth C. Arnold
Harvard CS
Cambridge, MA

Kai-Wei Chang
University of California
Los Angeles, CA

Adam T. Kalai
Microsoft Research
Cambridge, MA

Abstract

Mobile devices use language models to suggest words and phrases for use in text entry. Traditional language models are based on contextual word frequency in a static corpus of text. However, certain types of phrases, when offered to writers as suggestions, may be systematically chosen more often than their frequency would predict. In this paper, we propose the task of generating suggestions that writers accept, a related but distinct task to making accurate predictions. Although this task is fundamentally interactive, we propose a counterfactual setting that permits offline training and evaluation. We find that even a simple language model can capture text characteristics that improve acceptability.

1 Introduction

Intelligent systems help us write by proactively suggesting words or phrases while we type. These systems often build on a language model that picks *most likely* phrases based on previous words in context, in an attempt to increase entry speed and accuracy. However, recent work (Arnold et al., 2016) has shown that writers appreciate suggestions that have creative wording, and can find phrases suggested based on frequency alone to be boring. For example, at the beginning of a restaurant review, "I love this place" is a reasonable *prediction*, but a review *writer* might prefer a suggestion of a much less likely phrase such as "This was truly a wonderful experience"—they may simply not have thought of this more enthusiastic phrase. Figure 1 shows another example.

We propose a new task for NLP research: generate *suggestions* for writers. Doing well at this task requires innovation in language generation but also

Figure 1: We adapt a language model to offer *suggestions* during text composition. In above example, even though the middle suggestion is predicted to be about 1,000 times more likely than the one on the right, a user prefers the right one.

interaction with people: suggestions must be evaluated by presenting them to actual writers. Since writing is a highly contextual creative process, traditional batch methods for training and evaluating human-facing systems are insufficient: asking someone whether they think something would make a good suggestion in a given context is very different from presenting them with a suggestion in a natural writing context and observing their response. But if evaluating every proposed parameter adjustment required interactive feedback from writers, research progress would be slow and limited to those with resources to run large-scale writing experiments.

In this paper we propose a hybrid approach: we maintain a natural human-centered objective, but introduce a proxy task that provides an unbiased estimate of expected performance on human evaluations. Our approach involves developing a *stochastic* baseline system (which we call the *reference policy*), logging data from how writers interact with it, then estimating the performance of candidate policies by comparing how they would behave

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 49–54,
Taipei, Taiwan, November 27 – December 1, 2017

with how the reference policy did behave in the contexts logged. As long as the behavior of the candidate policy is not too different from that of the reference policy (in a sense that we formalize), this approach replaces complex human-in-the-loop evaluation with a simple convex optimization problem.

This paper demonstrates our approach: we collected data of how humans use suggestions made by a reference policy while writing reviews of a well-known restaurant. We then used logged interaction data to optimize a simple discriminative language model, and find that even this simple model generates better suggestions than a baseline trained without interaction data. We also ran simulations to validate the estimation approach under a known model of human behavior.

Our contributions are summarized below:

- We present a new NLP task of *phrase suggestion* for writing.[1]
- We show how to use counterfactual learning for goal-directed training of language models from interaction data.
- We show that a simple discriminative language model can be trained with offline interaction data to generate better suggestions in unseen contexts.

2 Related Work

Language models have a long history and play an important role in many NLP applications (Sordoni et al., 2015; Rambow et al., 2001; Mani, 2001; Johnson et al., 2016). However, these models do not model human preferences from interactions. Existing deployed keyboards use n-gram language models (Quinn and Zhai, 2016; Kneser and Ney, 1995), or sometimes neural language models (Kim et al., 2016), trained to predict the next word given recent context. Recent advances in language modeling have increased the accuracy of these predictions by using additional context (Mikolov and Zweig, 2012). But as argued in Arnold et al. (2016), these increases in accuracy do not necessarily translate into better suggestions.

The difference between suggestion and prediction is more pronounced when showing phrases rather than just words. Prior work has extended predictive language modeling to phrase prediction (Nandi and Jagadish, 2007) and sentence completion (Bickel et al., 2005), but do not directly model human preferences. Google's "Smart Reply" email response suggestion system (Kannan et al., 2016) avoids showing a likely predicted response if it is too similar to one of the options already presented, but the approach is heuristic, based on a priori similarity. Search engine query completion also generates phrases that can function as suggestions, but is typically trained to predict what query is made (e.g., Jiang et al. (2014)).

3 Counterfactual Learning for Generating Suggestions

We consider the task of generating good words and phrases to present to writers. We choose a pragmatic quality measure: a suggestion system is good if *it generates suggestions that writers accept*. Let h denote a suggestion system, characterized by $h(y|x)$, the probability that h will suggest the word or phrase y when in context x (e.g., words typed so far).[2] We consider deploying h in an interactive interface such as Figure 1, which suggests phrases using a familiar predictive typing interface. Let δ denote a reward that a system receives from that interaction; in our case, the number of words accepted.[3] We define the overall quality of a suggestion system by its expected reward $E[\delta]$ over all contexts.

Counterfactual learning allows us to evaluate and ultimately learn models that differ from those that were deployed to collect the data, so we can deploy a single model and improve it based on the data collected (Swaminathan and Joachims, 2015). Intuitively, if we deploy a model h_0 and observe what actions it takes and what feedback it gets, we could improve the model by making it more likely to suggest the phrases that got good feedback.

Suppose we deploy a *reference model*[4] h_0 and log a dataset

$$\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\}$$

of contexts (words typed so far), actions (phrases suggested), rewards, and propensities respectively, where $p_i \equiv h_0(y_i|x_i)$. Now consider deploying an alternative model h_θ (we will show an example as

[1] Code and data are available at https://github.com/kcarnold/counterfactual-lm.

[2] Our notation follows Swaminathan and Joachims (2015) but uses "reward" rather than "loss." Since $h(y|x)$ has the form of a contextual language model, we will refer to it as a "model."

[3] Our setting admits alternative rewards, such as the speed that a sentence was written, or an annotator's rating of quality.

[4] Some other literature calls h_0 a *logging policy*.

Eq. (1) below). We can obtain an unbiased estimate of the reward that h_θ would incur using importance sampling:

$$\hat{R}(h_\theta) = \frac{1}{n}\sum_{i=1}^{n} \delta_i h_\theta(y_i|x_i)/p_i.$$

However, the variance of this estimate can be unbounded because the importance weights $h_\theta(y_i|x_i)/p_i$ can be arbitrarily large for small p_i. Like Ionides (2008), we clip the importance weights to a maximum M:

$$\hat{R}^M(h) = \frac{1}{n}\sum\nolimits_{i=1}^{n} \delta_i \min\left\{M, h_\Theta(y_i|x_i)/p_i\right\}.$$

The improved model can be learned by optimizing

$$\hat{h_\theta} = \text{argmax}_h\, \hat{R}^M(h).$$

This optimization problem is convex and differentiable; we solve it with BFGS. [5]

4 Demonstration Using Discriminative Language Modeling

We now demonstrate how counterfactual learning can be used to evaluate and optimize the acceptability of suggestions made by a language model. We start with a traditional predictive language model h_0 of any form, trained by maximum likelihood on a given corpus.[6] This model can be used for generation: sampling from the model yields words or phrases that match the frequency statistics of the corpus. However, rather than offering representative samples from h_0, most deployed systems instead sample from $p(w_i) \propto h_0(w_i)^{1/\tau}$, where τ is a "temperature" parameter; $\tau = 1$ corresponds to sampling based on p_0 (soft-max), while $\tau \to 0$ corresponds to greedy maximum likelihood generation (hard-max), which many deployed keyboards use (Quinn and Zhai, 2016). The effect is to skew the sampling distribution towards more probable words. This choice is based on a heuristic assumption that writers desire more probable suggestions; what if writers instead find common phrases to be overly cliché and favor more descriptive phrases? To capture these potential effects, we add features that can emphasize various characteristics of the generated text, then use counterfactual learning to assign weights to those features that result in suggestions that writers prefer.

`LM` weight = 1, all other weights zero:
i didn't see a sign for; i am a huge sucker for
`LM` weight = 1, `long-word` bonus = 1.0:
another restaurant especially during sporting events
`LM` weight = 1, `POS` adjective bonus = 3.0:
great local bar and traditional southern

Table 1: Example phrases generated by the log-linear language model under various parameters. The context is the beginning-of-review token; all text is lowercased. Some phrases are not fully grammatical, but writers can accept a prefix.

We consider locally-normalized log-linear language models of the form

$$h_\theta(y|x) = \prod_{i=1}^{|y|} \frac{\exp \theta \cdot f(w_i|c, w_{[:i-1]})}{\sum_{w'} \exp \theta \cdot f(w'|c, w_{[:i-1]})}, \quad (1)$$

where y is a phrase and $f(w_i|x, w_{[:i-1]})$ is a feature vector for a candidate word w_i given its context x. ($w_{[:i-1]}$ is a shorthand for $\{w_1, w_2, \ldots w_{i-1}\}$.) Models of this form are commonly used in sequence labeling tasks, where they are called Max-Entropy Markov Models (McCallum et al., 2000). Our approach generalizes to other models such as conditional random fields (Lafferty et al., 2001).

The feature vector can include a variety of features. By changing feature weights, we obtain language models with different characteristics. To illustrate, we describe a model with three features below. The first feature (`LM`) is the log likelihood under a base 5-gram language model $p_0(w_i|c, w_{[:i-1]})$ trained on the Yelp Dataset[7] with Kneser-Ney smoothing (Heafield et al., 2013). The second and third features "bonus" two characteristics of w_i: `long-word` is a binary indicator of long word length (we arbitrarily choose ≥ 6 letters), and `POS` is a one-hot encoding of its most common POS tag. Table 1 shows examples of phrases generated with different feature weights.

Note that if we set the weight vector to zero except for a weight of $1/\tau$ on `LM`, the model reduces to sampling from the base language model with "temperature" τ. The fitted model weights of the log-linear model in our experiments is shown in supplementary material.

[5] We use the BFGS implementation in SciPy.

[6] The model may take any form, but n-gram (Heafield et al., 2013) and neural language models (e.g., (Kim et al., 2016)) are common, and it may be unconditional or conditioned on some source features such as application, document, or topic context.

[7] `https://www.yelp.com/dataset_challenge`; we used only restaurant reviews

Reference model h_0. In counterfactual estimation, we deploy one reference model h_0 to learn another $\hat{h}$—but weight truncation will prevent $\hat{h}$ from deviating too far from h_0. So h_0 must offer a broad range of types of suggestions, but they must be of sufficiently quality that some are ultimately chosen. To balance these concerns, we use temperature sampling with a temperature $\tau = 0.5$):

$$\frac{p_0(w_i|c, w_{[:i-1]})^{1/\tau}}{\sum_w p_0(w|c, w_{[:i-1]})^{1/\tau}}.$$

We use our reference model h_0 to generate 6-word suggestions one word at a time, so p_i is the product of the conditional probabilities of each word.

4.1 Simulation Experiment

We present an illustrative model of suggestion acceptance behavior, and simulate acceptance behavior under that model to validate our methodology. Our method successfully learns a suggestion model fitting writer preference.

Desirability Model. We model the behavior of a writer using the interface in Fig. 1, which displays 3 suggestions at a time. At each timestep i they can choose to accept one of the 3 suggestions $\{s_j^i\}_{j=1}^3$, or reject the suggestions by tapping a key. Let $\{p_j^i\}_{j=1}^3$ denote the likelihood of suggestion s_j^i under a predictive model, and let $p_\emptyset^i = 1 - \sum_{j=1}^3 p_j^i$ denote the probability of any other word. Let a_j^i denote the writer's probability of choosing the corresponding suggestion, and a_j^i denote the probability of rejecting the suggestions offered. If the writer decided exactly what to write before interacting with the system and used suggestions for optimal efficiency, then a_j^i would equal p_j^i. But suppose the writer finds certain suggestions *desirable*. Let D_j^i give the desirability of a suggestion, e.g., D_j^i could be the number of long words in suggestion s_j^i. We model their behavior by adding the desirabilities to the log probabilities of each suggestion:

$$a_j^{(i)} = p_j^{(i)} \exp(D_j^{(i)})/Z^{(i)},\ a_\emptyset^{(i)} = p_\emptyset^{(i)}/Z^{(i)}$$

where $Z^{(i)} = 1 - \sum_j p_j^{(i)}(1 - \exp(D_j^{(i)}))$. The net effect is to move probability mass from the "reject" action $a_\emptyset^i$ to suggestions that are close enough to what the writer wanted to say but desirable.

Experiment Settings and Results. We sample 10% of the reviews in the Yelp Dataset, hold them out from training h_0, and split them into an equal-sized training set and test set. We randomly sample suggestion locations from the training set. We cut off that phrase and pretend to retype it. We generate three phrases from the reference model h_0, then allow the simulated author to pick one phrase, subject to their preference as modeled by the desirability model. We learn a customized language model and then evaluate it on an additional 500 sentences from the test set.

For an illustrative example, we set the desirability D to the number of long words (≥ 6 characters) in the suggestion, multiplied by 10. Figure 3 shows that counterfactual learning quickly finds model parameters that make suggestions that are more likely to be accepted, and the counterfactual estimates are not only useful for learning but also correlate well with the actual improvement. In fact, since weight truncation (controlled by M) acts as regularization, the counterfactual estimate consistently *underestimates* the actual reward.

4.2 Experiments with Human Writers

We recruited 74 workers through MTurk to write reviews of *Chipotle Mexican Grill* using the interface in Fig 1 from Arnold et al. (2016). For the sake of simplicity, we assumed that all human writers have the same preference. Based on pilot experiments, Chipotle was chosen as a restaurant that many crowd workers had dined at. User feedback was largely positive, and users generally understood the suggestions' intent. The users' engagement with the suggestions varied greatly—some loved the suggestions and their entire review consisted of nearly only words entered with suggestions while others used very few suggestions. Several users reported that the suggestions helped them select words to write down an idea or also gave them ideas of what to write. We did not systematically enforce quality, but informally we find that most reviews written were grammatical and sensible, which indicates that participants evaluated suggestions before taking them. The dataset contains 74 restaurant reviews typed with phrase suggestions. The mean word count is 69.3, std=25.70. In total, this data comprises 5125 words, along with almost 30k suggestions made (including mid-word).

Estimated Generation Performance. We learn an improved suggestion model by the estimated expected reward ($\hat{R}^M$). We fix $M = 10$ and evaluate the performance of the learned parameters on held-

i love this place. the food is good. it's a little expensive, but the food is so much more than that. and i love the people that work there. the burritos are huge and packed with flavor. i got one with chicken and beef and extra quac. it tasted fresh and i couldn't even finish it all as it was huge! i love the location and the atmosphere was great. i will definitely come back to try something different!

i hate spicy food but for some reason i love the flavor of chipotle chiles in any form, so i loooove chipotle. i have been nervous about eating here lately because of the food poisoning scandals but thankfully i have not had any problems! i alway order the burrito bowls and the portions are huge! service is just mediocre but not bad and you cant expect too much from a chain restaurant. overall i would give chipotle four stars.

Figure 2: Example reviews. A colored background indicates that the word was inserted by accepting a suggestion. Consecutive words with the same color were inserted as part of a phrase.

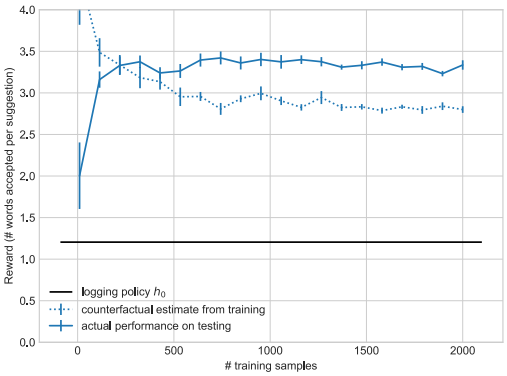

Figure 3: We simulated learning a model based on the behavior of a writer who prefers long words, then presented suggestions from that learned model to the simulated writer. The model learned to make desirable predictions by optimizing the counterfactual estimated reward. Regularization causes that estimate to be conservative; the reward actually achieved by the model exceeded the estimate.

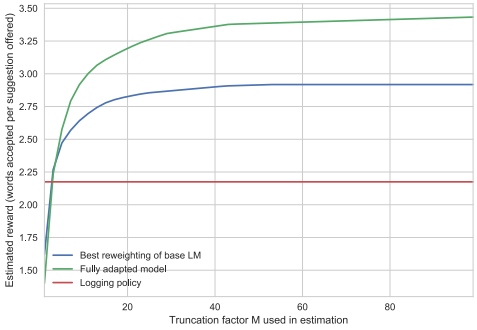

Figure 4: The customized model consistently improves expected reward over baselines (reference LM, and the best "temperature" reweighting LM) in held-out data. Although the result is an estimated using weight truncation at M, the improvement holds for all reasonable M.

out data using 5-fold cross-validation. Figure 4 shows that while the estimated performance of the new model does vary with the M used when estimating the expected reward, the relationships are consistent: the fitted model consistently receives the highest expected reward, followed by an ablated model that can only adjust the temperature parameter τ, and both outperform the reference model (with $\tau = 1$). The fitted model weights suggest that the workers seemed to prefer long words and pronouns, and eschewed punctuation.

5 Discussion

Our model assumed all writers have the same preferences. Modeling variations between writers, such as in style or vocabulary, could improve performance, as has been done in other domains (e.g., Lee et al. (2017)). Each review in our dataset was written by a different writer, so our dataset could be used to evaluate online personalization approaches.

Our task of crowdsourced reviews of a single restaurant may not be representative of other tasks or populations of users. However, the predictive language model is a replaceable component, and a stronger model that incorporates more context (e.g., Sordoni et al. (2015)) could improve our baselines and extend our approach to other domains.

Future work can improve on the simple discriminative language model presented here to increase grammaticality and relevance, and thus acceptability, of the suggestions that the customized language models generate.

Acknowledgements Kai-Wei Chang was supported in part by National Science Foundation Grant IIS-1657193. Part of the work was done while Kai-Wei Chang and Kenneth C. Arnold visited Microsoft Research, Cambridge.

References

Kenneth C. Arnold, Krzysztof Z. Gajos, and Adam T. Kalai. 2016. On suggesting phrases vs. predicting

words for mobile text composition. In *Proceedings of UIST '16*.

Steffen Bickel, Peter Haider, and Tobias Scheffer. 2005. Learning to complete sentences. In *Machine Learning: ECML 2005*, pages 497–504. Springer.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria.

Edward L Ionides. 2008. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311.

Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 445–454, New York, NY, USA. ACM.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 282–289.

Hung-Yi Lee, Bo-Hsiang Tseng, Tsung-Hsien Wen, Yu Tsao, Hung-Yi Lee, Bo-Hsiang Tseng, Tsung-Hsien Wen, and Yu Tsao. 2017. Personalizing recurrent-neural-network-based language model by social network. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 25(3):519–530.

Inderjeet Mani. 2001. *Automatic Summarization*, volume 3 of *Natural Language Processing*. John Benjamins Publishing Company, Amsterdam/Philadelphia.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *SLT*, pages 234–239.

Arnab Nandi and HV Jagadish. 2007. Effective phrase prediction. In *Proceedings of the 33rd international conference on Very large data bases*, pages 219–230. VLDB Endowment.

Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 83–88.

Owen Rambow, Srinivas Bangalore, and Marilyn Walker. 2001. Natural language generation in dialog systems. In *Proceedings of the first international conference on Human language technology research*, pages 1–4.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 939–941. International World Wide Web Conferences Steering Committee.