Automatically Learning Data Augmentation Policies for Dialogue Tasks

Tong Niu   Mohit Bansal
UNC Chapel Hill
{tongn, mbansal}@cs.unc.edu

Abstract

Automatic data augmentation (AutoAugment) (Cubuk et al., 2019) searches for optimal perturbation policies via a controller trained using performance rewards of a sampled policy on the target task, hence reducing data-level model bias. While being a powerful algorithm, their work has focused on computer vision tasks, where it is comparatively easy to apply imperceptible perturbations without changing an image's semantic meaning. In our work, we adapt AutoAugment to automatically discover effective perturbation policies for natural language processing (NLP) tasks such as dialogue generation. We start with a pool of atomic operations that apply subtle semantic-preserving perturbations to the source inputs of a dialogue task (e.g., different POS-tag types of stopword dropout, grammatical errors, and paraphrasing). Next, we allow the controller to learn more complex augmentation policies by searching over the space of the various combinations of these atomic operations. Moreover, we also explore conditioning the controller on the source inputs of the target task, since certain strategies may not apply to inputs that do not contain that strategy's required linguistic features. Empirically, we demonstrate that both our input-agnostic and input-aware controllers discover useful data augmentation policies, and achieve significant improvements over the previous state-of-the-art, including trained on manually-designed policies.

1 Introduction

Data augmentation aims at teaching invariances to a model so that it generalizes better outside the training set distribution. Recently, there has been substantial progress in Automatic Data Augmentation (*AutoAugment*) for computer vision (Cubuk et al., 2019). This algorithm searches for optimal perturbation policies via a controller trained with reinforcement learning, where its reward comes

from training the target model with data perturbed by the sampled augmentation policy. Each policy consists of 5 sub-policies sampled randomly during training, and each sub-policy consists of 2 operations applied in sequence. These operations are semantic-preserving image processing functions such as *translation* and *rotation*.

We adapt AutoAugment to NLP tasks, where the operations are subtle, semantic-preserving text perturbations. To collect a pool of such operations, the first challenge we face is that the discrete nature of text makes it less straightforward to come up with semantic-preserving perturbations. We thus employ as a starting point Should-Not-Change strategies (equivalent to *operations* in this paper) proposed by Niu and Bansal (2018) which are shown to improve their dialogue task performance when trained on data perturbed by these strategies. Importantly, we next divide their operations into several smaller ones (such as Grammar Errors divided into Singular/Plural Errors and Verb Inflection Errors) and also add a new operation Stammer (word repetition). This modification provides a much larger space of operation combinations for the model to explore from, so that it could potentially learn more complex and nuanced augmentation policies. Figure 1 shows a sub-policy containing two operations. It first paraphrases 2 tokens with probability 0.7 and then introduces 1 grammar error with probability 0.4.[1]

We choose the dialogue generation task based on the Ubuntu Dialogue Corpus (Lowe et al., 2015) because as opposed to Natural Language Inference and Question Answering tasks, real-world dialogue datasets more naturally afford such perturbation-style human errors (i.e., contain more noise), and thus are inherently compatible with

---

[1]Our code and sampled augmented data is publicly available at: https://github.com/WolfNiu/AutoAugDialogue. The learned policies are presented in Table 4.
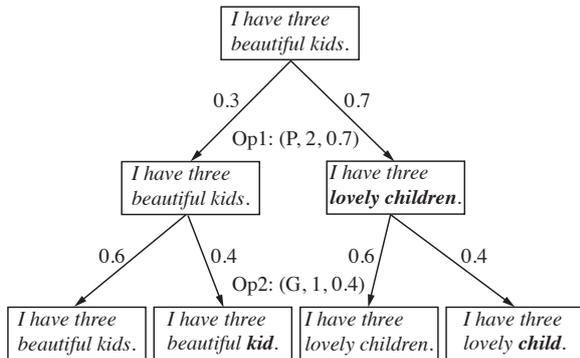
Figure 1: Example of a sub-policy applied to a source input. The first operation (Paraphrase, 2, 0.7) paraphrases the input twice with probability 0.7; the second operation (Grammar Errors, 1, 0.4) inserts 1 grammar error with probability 0.4. Thus there are at most 4 possible outcomes from each sub-policy.

a variety of artificial perturbations. Empirically, we show that our controller can self-learn policies that achieve state-of-the-art performance on Ubuntu, even compared with very strong baselines such as the best manually-designed policy in Niu and Bansal (2018). We also verify this result through human evaluation to show that our model indeed learns to generate higher-quality responses. We next analyze the best-learned policies and observe that the controller prefers to sample operations which work well on their own as augmentation policies, and then combines them into stronger policy sequences. Lastly, observing that some operations require the source inputs to have certain linguistic features (e.g., we cannot apply Stopword Dropout to inputs that do not contain stopwords), we also explore a controller that conditions on the source inputs of the target dataset, via a sequence-to-sequence (seq2seq) controller. We show that this input-aware model performs on par with the input-agnostic one (where the controller outputs do not depend on the source inputs), and may need more epochs to expose the model to the many diverse policies it generates. We also present selected best policies to demonstrate that the seq2seq controller can sometimes successfully attend to the source inputs.

2   Related Works

There has been extensive work that employs data augmentation in both computer vision (Simard et al., 2003; Krizhevsky et al., 2012; Cireşan et al., 2012; Wan et al., 2013; Sato et al., 2015; DeVries and Taylor, 2017; Tran et al., 2017; Lemley et al.,

2017) and NLP (Fürstenau and Lapata, 2009; Sennrich et al., 2016; Wang and Yang, 2015; Zhang et al., 2015; Jia and Liang, 2016; Kim and Rush, 2016; Hu et al., 2017; Xu et al., 2017; Xia et al., 2017; Silfverberg et al., 2017; Kafle et al., 2017; Hou et al., 2018; Wang et al., 2018).

Automatic data augmentation is addressed via the AutoAugment algorithm proposed by Cubuk et al. (2019), which uses a hypernetwork (in our case, a controller) to train the target model, an approach inspired by neural architecture search (Zoph et al., 2017). Previous works have also adopted Generative Adversarial Networks (Goodfellow et al., 2014) to either directly generate augmented data (Tran et al., 2017; Sixt et al., 2018; Antoniou et al., 2017; Zhu et al., 2017; Mun et al., 2017; Wang and Perez, 2017), or generate augmentation strategies (Ratner et al., 2017). These approaches produce perturbations through continuous hidden representations. Motivated by the fact that our pool of candidate perturbations are discrete in nature, we identify AutoAugment as a more proper base model to use, and adapt it linguistically to the challenging task of generative dialogue tasks. Our work closely follows Niu and Bansal (2018) to obtain a pool of candidate operations. Although their work also used combinations of operations for data augmentation, their best model is manually designed, training on each operation in sequence, while our model automatically discovers more nuanced and detailed policies that have not only the operation types but also the intensity (the number of changes) and probability.

3   Model

AutoAugment Architecture: As shown in Figure 2, our AutoAugment model consists of a controller and a target model (Cubuk et al., 2019). The controller first samples a policy that transforms the original data to augmented data, on which the target model trains. After training, the target model is evaluated to obtain the performance on the validation set. This performance is then fed back to the controller as the reward signal. Figure 3 illustrates the details of input-agnostic and input-aware controllers. The former corresponds to the lower half of the figure. It consists of a single decoder that samples each operation in sequence. An operation is defined by 3 parameters: *Operation Type*, *Number of Changes* (the maximum number of times allowed to perform the operation, which
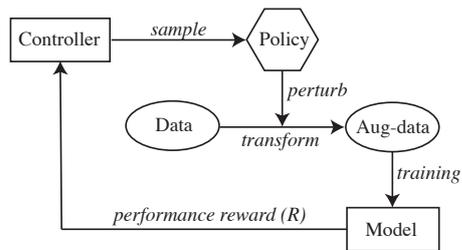
Figure 2: The controller samples a policy which is used to perturb the training data. After training on the augmented inputs, the model feeds the performance as reward back to the controller.

is a discrete equivalence of the continuous *Magnitude* in (Cubuk et al., 2019)), and the *Probability* of applying that operation. The input-aware controller corresponds to the whole figure, i.e., we novelly add in an encoder that takes as inputs the source of the training data, making it a seq2seq model. Since for each source input, there may be a different set of perturbations that are most suitable to it, our input-aware controller aims at providing customized operations for each training example.

Search Space: Following Niu and Bansal (2018), our pool of operations contains Random Swap, Stopword Dropout, Paraphrase, Grammar Errors, and Stammer, which cover a substantial enough search space of real-world noise in text.[2] To allow the controller to learn more nuanced combinations of operations, we further divide Stopword Dropout into 7 categories: Noun, Adposition, Pronoun, Adverb, Verb, Determiner, and Other, and divide Grammar Errors into Noun (plural/singular confusion) and Verb (verb inflected/base form confusion). For Stopword Dropout, we chose the first six categories because they are the major universal POS tags in the set closed-class words. We perform this subdivision also because different categories of an operation can influence the model to different extents. Suppose the original utterance is "*What is the offer?*", dropping the interrogative pronoun "*what*" is more semantic-changing than dropping the determiner "*the*." Our pool thus consists of 12 operations. For generality, we do not distinguish in advance which operation alone is effective as an augmentation policy on the target dataset, but rather let the controller figure it out. Moreover, it is possible that an operation alone is not effective, but when applied in sequence with another operation, they collectively teach the
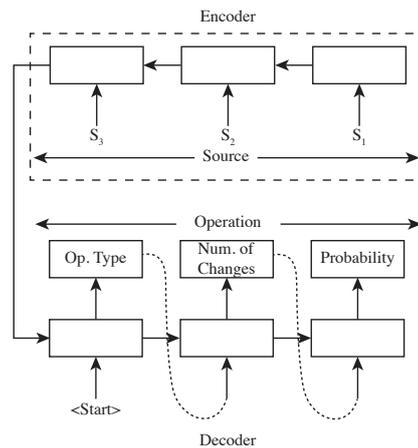

Figure 3: AutoAugment controller. An input-agnostic controller corresponds to the lower part of the figure. It samples a list of operations in sequence. An input-aware controller additionally has an encoder (upper part) that takes in the source inputs of the data.

model a new pattern of invariance in the data.

Each policy consists of 4 sub-policies chosen randomly during training, each sub-policy consists of 2 operations,[3] and each operation has 3 hyper-parameters. We discretize the Probability of applying an operation into 10 uniformly-spaced values starting from 0.1 and let the number of replaces range from 1 to 4. Thus, the search space size is $(12 \times 4 \times 10)^{(2 \times 4)} = 2.82 \times 10^{21}$. 2 operations allows more perturbations than exerted by a single strategy: in reality a sentence may contain several types of noise.

Search Algorithm: We use REINFORCE, a policy gradient method (Williams, 1992; Sutton et al., 2000) to train the controller. At each step, the decoder samples a token and feeds it into the next step. Since each policy consists of 4 sub-policies, each sub-policy contains 2 operations, and each operation is defined by 3 tokens (Operation Type, Number of Changes, Probability), the total number of steps for each policy is $4 \times 2 \times 3 = 24$. Sampling multiple sub-policies to form one policy provides us with a less biased estimation of the controller performance. We sample these 4 sub-policies at once rather than sample the controller 4 times to reduce repetition – the controller needs to keep track of what policies it has already sampled. To obtain the reward for the controller, we train the target model with the augmented data and obtain its validation set performance. We calculate a weighted average of two F1s (Activity and Entity)

---

[2]We did not use their Generative Paraphrasing where *number of changes* (which signifies the intensity of the policy) does not apply to this operation, because this operation can generate only one possible paraphrase for each input.

[3]Having 2 operations allows more perturbations than exerted by a single strategy: in reality a sentence may contain several types of noise.

as the reward R since both are important complementary measures of an informative response, as discussed in Serban et al. (2017a).[4] We use the reinforcement loss following Ranzato et al. (2016) and an exponential moving average baseline to reduce training variance. At the end of the search, we use the best policy to train the target model from scratch and evaluate on the test set.

4 Experimental Setup

Dataset and Model: We investigate Variational Hierarchical Encoder-Decoder (VHRED) (Serban et al., 2017b) on the troubleshooting Ubuntu Dialogue task (Lowe et al., 2015).[5] For this short paper, we focus on this dataset because it has a well-established F1 metric among dialogue tasks.[6] In future work, we plan to apply the idea to other datasets and NLP tasks since the proposed model is not specific to the Ubuntu dataset. We hypothesize that most web/online datasets that have regular human errors/noise (e.g. Twitter Dialogue Dataset (Serban et al., 2017b) and multiple movie corpora (Serban et al., 2016)) would be suitable for our framework.

Training Details: We employ code[7] from Niu and Bansal (2018) to reproduce the VHRED baseline results and follow methods described in Zoph et al. (2017) to train the controller. For details, please refer to their Appendix A.2 (Controller Architecture) and A.3 (Training of the Controller).

We adopt the following method to speed up controller training and hence facilitate scaling.[8] We let the target model resume from the converged baseline checkpoint and train on the perturbed data for 1 epoch. During testing, we use the policy that achieves the highest weighted average F1 score to train the final model for 1 epoch. For the All-operations model (which corresponds to the All-Should-Not-Change model in Niu and Bansal (2018)), we train on each operation (without subdivisions for Stopword Dropout and Grammar Er-

	Activity F1	Entity F1
LSTM	1.18	0.87
HRED	4.34	2.22
VHRED	4.63	2.53
VHRED (w/ attn.)	5.94	3.52
All-operations	6.53	3.79
Input-aware	**7.04**	3.90
Input-agnostic	7.02	**4.00**

Table 1: Activity, Entity F1 results reported by previous work (rows 1-4 from Serban et al. (2017a)), and the All-operations and AutoAugment models.

	W	T	L	W - L
Input-agnostic vs. baseline	48	23	29	19
Input-aware vs. baseline	45	27	28	17
Input-agnostic vs. All-ops	43	27	30	13
Input-aware vs. All-ops	50	13	37	13

Table 2: Human evaluation results on comparisons among the baseline, All-operations, and the two AutoAugment models. **W**: Win, **T**: Tie, **L**: Loss.

rors) for 1 epoch in sequence.[9]

Automatic Evaluation: We follow Serban et al. (2017a) and evaluate on F1s for both activities (technical verbs) and entities (technical nouns).

Human Evaluation: We conducted human studies on MTurk. We compared each of the input-agnostic/aware models with the VHRED baseline and All-operations from Niu and Bansal (2018), where we followed the same setting. Each study contains 100 samples randomly chosen from the test set. The utterances were randomly shuffled and we only allowed US-located human evaluators with approval rate $> 98\%$, and at least $10,000$ approved HITs. More details are in the appendix.

5 Results and Analysis

Automatic Results: Table 1 shows that all data-augmentation approaches (last 3 rows) improve statistically significantly ($p < 0.01$) [10] over the strongest baseline VHRED (w/ attention). Moreover, our input-agnostic AutoAugment is statistically significantly ($p < 0.01$) better (on Activity and Entity F1) than the strong manual-policy All-operations model, while the input-aware model is stat. signif. ($p < 0.01$) better on Activity F1.[11]

Human Evaluation: In Table 2, both AutoAug-

---

[4]We use the weighted average F1 = Activity F1 + 5.94 / 3.52 ∗ Entity F1, where the weights are determined by the baseline results, for balance between the two F1s.

[5]http://www.iulianserban.com/Files/UbuntuDialogueCorpus.zip

[6]Serban et al. (2017a) found that F1 is particularly suited for the goal-oriented Ubuntu Dialogue Corpus based on manual inspection of the extracted activities and entities.

[7]https://github.com/WolfNiu/AdversarialDialogue

[8]It took us 3 days with 6 P100 GPUs to get our reported results for each of the input-aware and input-agnostic models. The VHRED model took 1.5 days to pre-train.

[9]We do not follow Cubuk et al. (2019) to train on a small portion of the data for many epochs because we empirically found that this approach decreases our task performance.

[10]Statistical significance is computed via the bootstrap test (Noreen, 1989) over 100K samples.

[11]Note that F1s are overall low (reported baselines in Table 1 are from Serban et al. (2017a)) because Ubuntu is semi-task-oriented; each context can have many valid responses containing different activities/entities from the ground-truth.

Perturbation Method	Policy	Source Inputs
Original Context	-	fresh install of crack of the day : gdm login → " can't access ACPI bla bla bla " __eou__ you don't want to be me ... __eou__ ah , it happened to you too ?
All-operations	(R , 4, 1.0)	fresh install **crack of** of the day : gdm login → " can't **ACPI access** bla bla bla " __eou__ you don't want **be to** me ... __eou__ ah , it happened **you to** too ?
Input-agnostic	(D$_v$, 3, 0.2) (R , 1, 0.5)	fresh install of crack of the day : **login gdm** → " can't access ACPI bla bla bla " __eou__ you **don't** want to **be** me ... __eou__ ah , it happened to you too ?
Input-aware	(S , 1, 0.8) (D$_v$, 2, 0.5)	fresh install of crack of the day : gdm login → " can't access ACPI bla bla **bla** bla " __eou__ you don't want to be me ... __eou__ ah , it happened to you too ?

Table 3: Comparisons of perturbed outputs among the three Augmentation models. Note that only the sampled/applied sub-policy is shown for each model. The dashed line separates the original and augmented source inputs. Perturbed tokens are boldfaced.

Sub-policy1	Sub-policy2	Sub-policy3	Sub-policy4
P, 1, 0.5	D$_v$, 3, 0.2	R, 3, 0.9	D$_p$, 2, 0.3
D$_{adv}$, 4, 0.4	R, 1, 0.5	D$_{adp}$, 1, 0.5	D$_{adp}$, 2, 0.1
D$_n$, 1, 0.8	D$_o$, 3, 1.0	P, 4, 0.4	G$_n$, 3, 0.3
G$_v$, 1, 0.9	D$_o$, 3, 0.1	S, 3, 0.4	R, 1, 0.2
D$_v$, 2, 0.5	D$_v$, 2, 0.7	S, 3, 0.5	P, 1, 1.0
R, 2, 0.2	G$_v$, 1, 0.9	D$_o$, 1, 0.5	G$_n$, 2, 0.6

Table 4: Top 3 policies on the validation set and their test performances. **Operations**: R=Random Swap, D=Stopword Dropout, P=Paraphrase, G=Grammar Errors, S=Stammer. **Universal tags**: n=noun, v=verb, p=pronoun, adv=adverb, adp=adposition.

ment models obtained significantly more net wins (last column) than the VHRED-attn baseline. They both outperform even the strong manual-policy All-operations model.

Policy Learned by the Input-Agnostic Controller: We present 3 best learned policies from the Ubuntu val set (Table 4). Although there is a probability of $7/12 = 58.3\%$ to sample one of the Stopword Dropout operations from our pool, all 3 learned policies show much more diversity on the operations they choose. This is also the case for the other two hyperparameters: *Number of Changes* varies from 1 to 4, and *Probability* varies from 0.1 to 0.9, which basically extend their whole search range. Moreover, all best policies include Random Swap, which agrees with the results in Niu and Bansal (2018).

Example Analysis of Perturbation Procedure in Generated Responses: We also present a selected example of perturbed source inputs from the three Augmentation models with their respective best policies in Table 3. First of all, the All-operations model is forced to use an operation (in this case Random Swap) with a fixed number of changes and a probability of 1.0, leading to less variation in the source inputs. On the other hand, our Input-agnostic AutoAugment model samples 3 Verb Dropout's followed by Random Swap.

Note that although the number of changes for the dropout is 3, there are only 2 verb stopwords in the utterance. Thus, it has to resort back to modifying only 2 tokens. The Input-aware model samples Stammer followed by 2 Verb Dropout's. Interestingly, it inserts an extra "*bla*" around other "*bla*'s" in the utterance. It also did not sample a policy that drops more than 2 verb stopwords (this operation is not applied due to its Probability parameter). These two observations indicate that the model can sometimes successfully attend to the source inputs to provide customized policies.

6 Conclusion and Future Work

We adapt AutoAugment to dialogue and extend its controller to a conditioned model. We show via automatic and human evaluations that our AutoAugment models learn useful augmentation policies which lead to state-of-the-art results on the Ubuntu task. Motivated by the promising success of our model in this short paper, we will apply it to other diverse NLP tasks in future work.

Acknowledgments

References

Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*.

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

International Conference on Learning Representations.

Dan Cireşan, Ueli Meier, and Jürgen Schmidhuber. 2012. Multi-column deep neural networks for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. 2019. AutoAugment: Learning augmentation policies from data. In *Proceedings of Conference on Computer Vision and Pattern Recognition*.

Terrance DeVries and Graham W Taylor. 2017. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*.

Hagen Fürstenau and Mirella Lapata. 2009. Semi-supervised semantic role labeling. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–228. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of COLING*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning, PMLR 70*, pages 1587–1596.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of Association for Computational Linguistics*.

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. 2017. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 198–202.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 1317–1327.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in neural information processing systems*, pages 1097–1105.

Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. 2017. Smart augmentation learning an optimal data augmentation strategy. In *Proceedings of IEEE Access*, volume 5, pages 5858–5869.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909*.

Seongkyu Mun, Sangwook Park, David K Han, and Hanseok Ko. 2017. Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane. *Detection and Classification of Acoustic Scenes and Events Workshop*, pages 93–97.

Tong Niu and Mohit Bansal. 2018. Adversarial over-sensitivity and over-stability strategies for dialogue models. In *Proceedings of Conference on Natural Language Learning*.

Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of International Conference on Learning Representations*.

Alexander J Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. 2017. Learning to compose domain-specific transformations for data augmentation. In *Proceedings of Advances in neural information processing systems*, pages 3236–3246.

Ikuro Sato, Hiroki Nishimura, and Kensuke Yokoi. 2015. Apac: Augmented pattern classification with neural networks. *arXiv preprint arXiv:1505.03229*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 86–96.

Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron C. Courville. 2017a. Multiresolution recurrent neural networks: An application to dialogue response generation. In *AAAI*, pages 3288–3294.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *The Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 3776–3784.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017b. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of AAAI*, pages 3295–3301.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99.

Patrice Y Simard, Dave Steinkraus, and John C Platt. 2003. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of International Conference on Document Analysis and Recognition*.

Leon Sixt, Benjamin Wild, and Tim Landgraf. 2018. Rendergan: Generating realistic labeled data. *Frontiers in Robotics and AI*, 5:66.

Richard S. Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press.

Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. 2017. A bayesian data augmentation approach for learning deep models. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2797–2806.

Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. 2013. Regularization of neural networks using dropconnect. In *Proceedings of International Conference on Machine Learning*, pages 1058–1066.

Jason Wang and Luis Perez. 2017. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Visual Recognition*.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of Empirical Methods in Natural Language Processing*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Proceedings of Reinforcement Learning*, pages 5–32. Springer.

Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. 2017. Dual supervised learning. In *Proceedings of ICML*, pages 3789–3798.

Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *Proceedings of Association for the Advancement of Artificial Intelligence*, pages 3358–3364.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of Advances in neural information processing systems*, pages 649–657.

Xinyue Zhu, Yifan Liu, Zengchang Qin, and Jiahong Li. 2017. Data augmentation in emotion classification using generative adversarial networks. *arXiv preprint arXiv:1711.00648*.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. 2017. Learning transferable architectures for scalable image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

A   Experimental Setup

Dataset and Model: We investigate Variational Hierarchical Encoder-Decoder (VHRED) (Serban et al., 2017b), a state-of-the-art dialogue model on the Ubuntu Dialogue Corpus (Lowe et al., 2015). Ubuntu is a troubleshooting dialogue dataset containing 1 million 2-person, multi-turn dialogues extracted from Ubuntu chat. The chat channels are used by customers to provide and receive technical support. We focus on the task of generating fluent, relevant, and goal-oriented responses. Following Niu and Bansal (2018), we apply additive attention mechanism (Bahdanau et al., 2015) to the source sequence while keeping the remaining VHRED architecture unchanged.

Automatic Evaluation: We follow Serban et al. (2017a) and evaluate the model on F1's for both activities (technical verbs, e.g., "boot", "delete") and entities (technical nouns, e.g., "root", "disk") computed by mapping the ground-truth and model responses to their corresponding activity-entity representations.

Human Evaluation: We conducted human studies on MTurk to evaluate the dialogue quality of generated responses from the investigated models. We compared each of the input-agnostic/aware models with All-operations and the baseline (i.e., 4 experiments in total). Studies involving the baseline are for sanity checks.