Pseudo Siamese Network for Few-shot Intent Generation

Congying Xia
cxia8@uic.edu
University of Illinois at Chicago
Chicago, IL, USA

Caiming Xiong
cxiong@salesforce.com
Salesforce Research
Palo Alto, CA, USA

Philip Yu
psyu@uic.edu
University of Illinois at Chicago
Chicago, IL, USA

ABSTRACT

Few-shot intent detection is a challenging task due to the scare annotation problem. In this paper, we propose a Pseudo Siamese Network (PSN) to generate labeled data for few-shot intents and alleviate this problem. PSN consists of two identical subnetworks with the same structure but different weights: an action network and an object network. Each subnetwork is a transformer-based variational autoencoder that tries to model the latent distribution of different components in the sentence. The action network is learned to understand action tokens and the object network focuses on object-related expressions. It provides an interpretable framework for generating an utterance with an action and an object existing in a given intent. Experiments on two real-world datasets show that PSN achieves state-of-the-art performance for the generalized few shot intent detection task.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Information extraction**.

KEYWORDS

Few-shot Learning, Intent Detection, Text Generation

ACM Reference Format:
Congying Xia, Caiming Xiong, and Philip Yu. 2021. Pseudo Siamese Network for Few-shot Intent Generation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3462995

1 INTRODUCTION

Intelligent assistants have gained great popularity recently. Companies are striving to deliver their products either on speaker devices such as Amazon Alexa, or smartphones such as Siri from Apple. To provide an intelligent conversational interface, these assistants need to understand the user's input correctly. Among all the natural language understanding tasks, intent detection is an important and essential one. It aims at understanding the goals underlying input utterances and classifying these utterances into different types of intents. For example, given an input utterance, "How's the weather in Chicago tomorrow?", the system needs to identify the intent is query weather. With the development of deep learning techniques, intent detection has achieved great success by formalizing it as a text classification task under the supervised learning paradigm [3, 25]. These works rely on a large amount of labeled data to train the intent detection model. Certain restrictions like requiring sufficient labeled examples for each class limit these models' ability to adapt to previously unseen classes promptly. Recently, researchers are interested in achieving decent performance with reduced human annotation and extending models' ability to detect new classes. Low-resource learning paradigms [22, 27] like Zero-shot learning [21] and Few-shot learning [13, 18, 19] have drawn a lot of attention recently. In this work, we focus on the task of identifying few-shot intents which only have a few labeled examples.

The bottleneck for identifying few-shot intents is the lack of annotations. If we can generate high-quality pseudo-labeled examples for these few-shot intents, we can effectively alleviate this issue and improve the performance. There are only a few previous works [11, 12, 16, 26] that try to augment the training data with generation methods and alleviate the scarce annotation problem. However, these models utilize simple neural networks with limited model capacity, like LSTMs [7], to do text generation. Furthermore, these methods do not consider the inner structure for an intent. Naturally, an intent can be defined as an action with an object [24]. For example, the intent of the input "wake me up at 7 am" is to set an alarm. This intent consists of an action "Set" and an object "Alarm". In this paper, we propose a Pseudo Siamese Network (PSN) that generates labeled examples for few-shot intents considering the inner structure of an intent. PSN consists of two identical subnetworks with the same structure but different weights: an action network and an object network. To utilize the powerful pre-trained language models and capture the latent distribution of sentences with different intents, we propose to use transformer-based [15] variational autoencoders [10] as the sub-networks to model different components in the sentences. The action network is learned to understand action tokens and the object network focuses on object-related expressions. During the inference, PSN generates an utterance with a given intent by controlling the action generation and the object generation separately in two subnetworks. It provides an interpretable framework for generating an utterance with an action and an object existing in a given intent.

To quantitatively evaluate the effectiveness of PSN for augmenting training data in low-resource intent detection, experiments are conducted for the generalized few-shot intent detection task (GFSID) [20]. GFSID is a more practical setting for few-shot intents. It not only considers the few-shot intents with a few labeled examples, but also includes existing intents with enough annotations. Formally, GFSID aims to discriminate a joint label space consisting of both existing many-shot intents and few-shot intents. In summary, the main contributions of our work are as follows. 1) We propose

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462995

a Pseudo Siamese Network to generate high-quality labeled data for few-shot intents and alleviate the scarce annotation problem. 2) PSN provides an interpretable framework for generating an utterance with an action and an object belonging to a given intent by controlling each part in a subnetwork. 3) Empirical experiments conducted on two real-world datasets show the effectiveness of our proposed model on the generalized few-shot intent detection task.

2 PSEUDO SIAMESE NETWORK

In this section, we introduce the details for the proposed Pseudo Siamese Network (PSN). As illustrated in Figure 1, PSN consists of two identical subnetworks: an action network and an object network. These two subnetworks have the same structure with different weights. Each subnetwork is utilized to model different components in the utterances. The action network is used to learn action tokens and the object network is proposed to focus on object-related expressions. Specifically, each subnetwork is a transformer-based variational autoencoder that consists of an encoder and a decoder. Each encoder and decoder are a stack of multiple transformer layers.

2.1 Input Representation

Each training instance consists of an input sentence and a corresponding intent. To capture the inner structure of the intent, we define the intent as a pair of an action y_a and an object y_o. Given an input sentence $s = (w_1, w_2, ..., w_n)$ with n tokens, we construct two text pairs and feed them separately into two subnetworks. We feed the action token together with the input sentence into the action network, while the object token and the input sentence are fed into the object network. To formalize the input for transformer-based models, we add a special start-of-sequence ([CLS]) token at the beginning of each input and a special end-of-sequence ([SEP]) token at the end of each sequence.

Formally, the input for the action network is formatted as ([CLS], y_a, [SEP], w_1, w_2, ..., w_n, [SEP]) and the input for the object network is ([CLS], y_o, [SEP], w_1, w_2, ..., w_n, [SEP]). The input of each subnetwork consists of two sentences. In this paper, we refer ([CLS], y_a, [SEP]) and ([CLS], y_o, [SEP]) to as S_1, and (w_1, w_2, ..., w_n, [SEP]) as S_2 in each subnetwork. For each input in the subnetwork, they are tokenized into subword units by WordPiece [17]. The input embeddings of a token sequence are represented as the sum of three embeddings: token embeddings, position embeddings [15], and segment embeddings [5]. These embeddings for input representation are shared between the action network and the object network.

2.2 Network Structure

The overall framework of Pseudo Siamese Network is illustrated in Figure 1. PSN consists of an action network and an object network. The action network has an action encoder and an action decoder while the object network has an object encoder and an object decoder. We will describe the encoders and the decoders separately in this section.

2.2.1 Encoders. Two encoders including the action encoder and the object encoder are contained in PSN. The action encoder encodes the action and the input sentence into a latent variable z_a while the object encoder encodes the object and the input sentence into a latent variable z_o. Multiple transformer layers [15] are utilized in the encoders. Each transformer layer models the self-attentions among all the tokens. For the l-th transformer layer, the output of a self-attention head A_l is computed via:

$$A_l = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{1}$$

where Q, K, V are queries, keys, and values projected from the previous layer H^{l-1} and parameterized by matrices $W_Q^l, W_K^l, W_V^l \in \mathbb{R}^{d_h \times d_k}$:

$$Q = H^{l-1}W_Q^l,\ K = H^{l-1}W_K^l,\ V = H^{l-1}W_V^l. \tag{2}$$

The embeddings for the [CLS] token that output from the last transformer layer in the encoder are used as the encoded sentence-level information. The encoded sentence-level information is denoted as e_a in the action encoder and e_o in the object encoder. e_o and e_a are encoded into z_a and z_o to model the distribution for the action and the object separately.

By modeling the true distributions, $p(z_a|x, y_a)$ and $p(z_o|x, y_o)$, using a known distribution that is easy to sample from [9], we constrain the prior distributions, $p(z_a|y_a)$ and $p(z_o|y_o)$, as multivariate standard Gaussian distributions. A reparametrization trick [10] is used to generate the latent vectors z_a and z_o separately. Gaussian parameters $(\mu_a, \mu_o, \sigma_a^2, \sigma_o^2)$ are projected from e_a and e_o:

$$\mu_a = e_a W_{\mu_a} + b_{\mu_a}, \tag{3}$$

$$\log(\sigma_a^2) = e_a W_{\sigma_a} + b_{\sigma_a}, \tag{4}$$

$$\mu_o = e_o W_{\mu_o} + b_{\mu_o}, \tag{5}$$

$$\log(\sigma_o^2) = e_o W_{\sigma_o} + b_{\sigma_o}, \tag{6}$$

where $W_{\mu_a}, W_{\mu_o}, W_{\sigma_a}, W_{\sigma_o} \in \mathbb{R}^{d_h \times d_h}$ and $b_{\mu_a}, b_{\mu_o}, b_{\sigma_a}, b_{\sigma_o} \in \mathbb{R}^{d_h}$. Noisy variables $\varepsilon_a \sim \mathcal{N}(0, \mathrm{I}), \varepsilon_o \sim \mathcal{N}(0, \mathrm{I})$ are utilized to sample z_a and z_o from the learned distribution:

$$z_a = \mu_a + \sigma_a \cdot \varepsilon_a, z_o = \mu_o + \sigma_o \cdot \varepsilon_o. \tag{7}$$

2.2.2 Decoders. The decoder utilizes latent variables together with labels to reconstruct the input sentence $p(s|z_a, z_o, y_a, y_o)$. As shown in Figure 1, the action decoder takes z_a, y_a, and the sentence $s = (w_1, w_2, ..., w_n)$ as the input while the input of the object decoder are z_o, y_o, and the sentence s. The label components (y_a, y_o) and the sentence s are embedded with an embedding layer. The embedding parameters are shared with the input representation.

To keep a fixed length and introduce the latent information z_a and z_o into the decoders, we replace the first [CLS] token with z_a and z_o in each sub-network. The decoders are also built with multiple transformer layers. Text generation is a sequential process that uses the left context to predict the next token. Inspired by [6] that utilizes specific self-attention masks to control what context the prediction conditions on, we apply the sequence-to-sequence attention mask proposed in Dong[6] in the decoders to simulate the left-to-right generation process. With the attention mask applied in the decoders, tokens in S_1 can only attend to tokens in S_1, while tokens in S_2 can attend to tokens in S_1 and all the left tokens in S_2. For the first tokens in two decoders, z_a and z_a, which hold latent information, they are only allowed to attend to themselves due to the vanishing latent variable problem. The latent information can

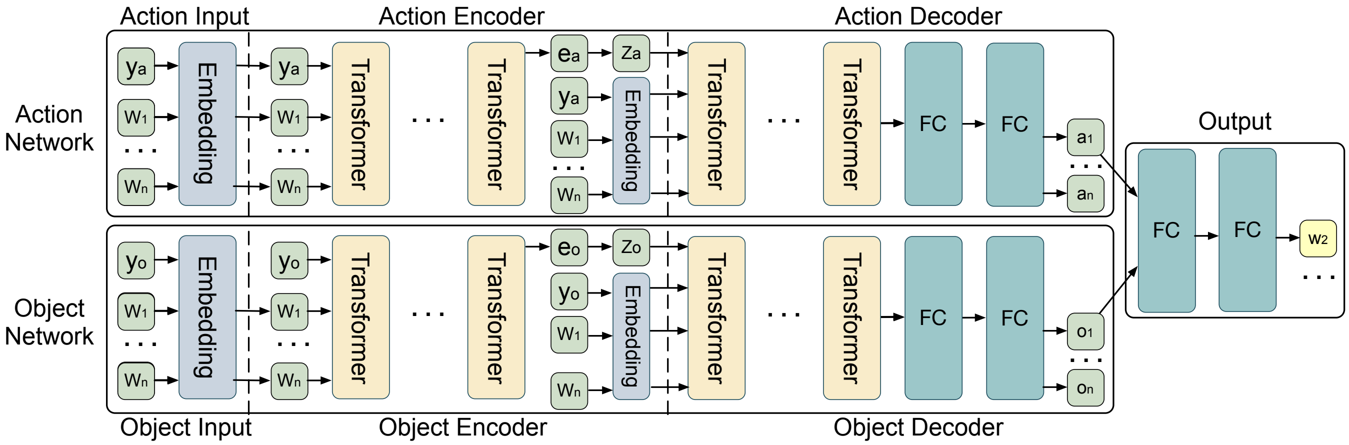

Figure 1: The overall framework of Pseudo Siamese Network. FC is short for Fully-Connected layers.

be overwhelmed by the information of other tokens when adapting VAE to natural language generators [28].

After the transformer layers in the decoders, we can obtain the embedding outputs for these two sequences: (z_a, y_a, [SEP], w_1, ..., w_n, [SEP]) and (z_o, y_o, [SEP], w_1, ..., w_n, [SEP]). To further increase the impact of the latent information and alleviate the vanishing latent variable problem, we concatenate the output embeddings of z_a to other token embeddings in the first sequence and concatenate z_o to other token embeddings in the second sequence. The hidden dimension increases to $2 \times d_h$ after the concatenation. To reduce the hidden dimension to d_h and get the embeddings to decode the vocabulary, two fully-connected (FC) layers followed by a layer normalization [1] are applied on top of the transformer layers. Gelu is used as the activation function in these two FC layers. For the token at position i in the sentence s, the output representation from the action decoder is denoted as a_i and o_i from the object decoder.

As shown in the output box of Figure 1, the outputs from action decoder and object decoder are fused together to predict the next token. An FC layer is used to fuse these outputs:

$$m_{i+1} = g(a_i W_a + o_i W_o + b), \tag{8}$$

where $W_o, W_a \in \mathbb{R}^{d_h \times d_h}$ and $b \in \mathbb{R}^{d_h}$ are parameters, and g is the GELU activation function. The fused embeddings m_{i+1} are used to predict the token at position $i+1$ with another FC layer. The inference process iteratively decodes the output till the [SEP] token is generated.

2.3 Loss Function

In the model, the loss function consists of two parts: the KL-divergence that regularize the prior distributions for two latent variables to be close to the Gaussian distributions and the reconstruction loss:

$$\begin{aligned}\mathcal{L} = &-\mathbb{E}_{q(z_a|x,y_a),q(z_o|x,y_o)}[\log p(x|z_a, z_o, y_a, y_o)] \\ &+\mathbb{D}_{\mathrm{KL}}[q(z_a|s, y_a), p(z_a|y_a)] \\ &+\mathbb{D}_{\mathrm{KL}}[q(z_o|s, y_o), p(z_o|y_o)].\end{aligned} \tag{9}$$

In the inference, utterances for few-shot intents are generated by sampling two latent variables, z_a and z_o, separately from multivariate standard Gaussian distributions. Beam search is applied to do the generation. To improve the diversity of the generated utterances, we sample the latent variables for s times and save the top k results for each time. These generated utterances are added to the original training datasets to alleviate the scare annotation problem.

Dataset	SNIPS-NLU	NLUED
Vocab Size	10,896	6,761
#Total Classes	7	64
#Few-shot Classes	2	16
#Few-shots / Class	1 or 5	1 or 5
#Training Examples	7,858	7,430
#Training Examples / Class	1571.6	155
#Test Examples	2,799	1,076
Average Sentence Length	9.05	7.68

Table 1: Data Statistics for SNIPS-NLU and NLUED. #Few-shot examples are excluded in the #Training Exampels. For NLUED, the statistics is reported for KFold_1.

3 EXPERIMENTS

3.1 Datasets

To demonstrate the effectiveness of our proposed model, we evaluate PSN on two real-word datasets for the generalized few-shot intent detection task: SNIPS-NLU [4] and NLU-Evaluation-Data (NLUED) [23]. These two datasets were collected to benchmark the performance of natural language understanding services offering customized solutions. Dataset details are illustrated in Table 1.

3.2 Baselines

We compare the proposed model with five baselines. 1) Prototypical Network [14] (PN) is a distance-based few-shot learning model. BERT-PN is a variation of PN by using BERT as the encoder, which is referred to as BERT-PN. 2) BERT. We over-sampled the few-shot intents for this baseline. 3) SVAE [2] is a variational autoencoder built with LSTMs. 4) CGT [8] adds a discriminator based on SVAE to classify the sentence attributes. 5) EDA [16] uses simple data augmentations rules for language transformation. We apply three rules in the experiment, including insert, delete and swap. 6) CG-BERT [20] is the first work that combines CVAE with BERT to do

	Seen	Unseen SNIPS-NLU 5-shot	H-Mean	Seen	Unseen NLUED 5-shot	H-Mean
BERT-PN	95.96 ± 1.13	86.03 ± 2.00	90.71 ± 1.19	83.41 ± 2.62	60.28 ± 4.19	69.93 ± 3.49
BERT	98.34 ± 0.10	81.82 ± 6.16	89.22 ± 3.74	**94.12 ± 0.89**	51.69 ± 3.19	66.67 ± 2.51
BERT + SVAE	**98.34 ± 0.06**	82.10 ± 4.06	89.49 ± 2.47	93.60 ± 0.63	54.03 ± 3.91	68.42 ± 3.06
BERT + CGT	98.32 ± 0.14	82.65 ± 4.31	89.78 ± 2.83	93.61 ± 0.63	54.70 ± 4.06	68.96 ± 3.17
BERT + EDA	98.09 ± 0.18	82.00 ± 3.47	89.30 ± 2.12	93.71 ± 0.64	57.22 ± 4.35	70.95 ± 3.35
BERT + CG-BERT	98.30 ± 0.17	86.89 ± 4.05	92.20 ± 2.32	93.80 ± 0.60	61.06 ± 4.29	73.88 ± 3.10
BERT + PSN	98.16 ± 0.12	**88.17 ± 1.19**	**92.89 ± 0.67**	92.82 ± 0.90	**64.16 ± 3.94**	**75.81 ± 2.87**

Table 2: Generalized few shot experiments with 5-shot setting on SNIPS-NLU and NLUED.

Query Alarm	
R1: what time is my alarm set for	G1: is my alarm set for seven am
R2: what time is my alarm set for tomorrow morning	G2: tell me the alarm for saturday morning
R3: tell me when it is five pm (**Set Alarm**)	B3: tell me when it is five pm
Recommendation Events	
R4: show latest events around new york	G4: what ' s the show around new york
R5: what are all the event in area	G5: check for all the event
R6: is there anything to do tonight	B6: what show is there anything to do tonight

Table 3: Generation examples from PSN. Rs are real examples, Gs are good generation examples and Bs are bad cases.

few-shot text generation. BERT is fine-tuned with the augmented training data for these generation baselines. The whole pipelines are referred to as BERT + SVAE, BERT + CGT, BERT + EDA and BERT + CG-BERT in Table 2.

For PSN, we use the first six layers in BERT-base to initialize the weights in the encoders transformer layers while the latter six layers are used to initialize the decoders. PSN is trained with a learning rate equal to 1e-5 in 100 epochs and each epoch has 1000 steps. The batch size is 16. New utterances are generated by sampling the latent variables $s = 10$ times and choosing the top $k = 30$ utterances.

3.3 Results

For SNIPS-NLU, the performance is calculated with the average and the standard deviation over 5 runs. The results on NLUED are reported over 10 folds. Three metrics are used to evaluate the model performances, including the accuracy on existing intents (Seen), the accuracy on few-shot intents (Unseen) together with their harmonic mean (H-mean) [20]. The harmonic mean is high only when the accuracy on both existing intents (Seen) and few-shot intents (Unseen) are high. As illustrated in Table 2, PSN achieves state-of-the-art performance on Unseen accuracy and H-mean and comparable performance on Seen accuracy. Compared to the few-shot learning baseline, BERT-PN, PSN improves the F1 score by 2.4% from 90.71% to 92.89% for the NULED 5-shot setting. Compared to other data augmentation baselines, we improve the best baseline CG-BERT by 2.6% from 73.88% to 75.81%. The improvement mainly stems from the high quality of the generated examples for few-shot intents, which leads to significantly increased Unseen accuracy and H-mean.

To evaluate the quality of the generated utterances and interpret how can PSN generate examples for few-shot intents, we show some examples generated by PSN. As illustrated in Table 3, PSN generates good examples by providing new slot values either for objects or new words for actions. For example, G1 generates "seven am" for the alarm object and G4 provides "the show" for the event object. Another type of augmentation comes from the action tokens. For example, G2 utilizes "tell me" for the "Query" action in the intent of "Query Alarm", while G5 generates "check" for recommendation. There are also bad cases like B3 that is generated for "Query Alarm" but comes from a similar intent "Set Alarm". The other type of bad case, like B6, has syntax errors.

4 CONCLUSIONS

In this paper, we propose a Pseudo Siamese Network (PSN) to generate labeled data for few-shot intents. PSN consists of two sub-networks (an action network and an object network) with the same structure but different weights. Each sub-network is a transformer-based variational autoencoder. They are trained to learn either the action or the object existing in the intent. It provides an interpretable framework for generating an utterance for a given intent. Experiments on two real-world datasets show that PSN achieves state-of-the-art performance for the generalized few shot intent detection task.

ACKNOWLEDGMENTS

We thank the reviewers for their valuable comments. This work is supported in part by NSF under grants III-1763325, III-1909323, and SaTC-1930941.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[2] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349* (2015).

[3] Yun-Nung Chen, Dilek Hakkani-Tür, Gökhan Tür, Jianfeng Gao, and Li Deng. 2016. End-to-End Memory Networks with Knowledge Carryover for Multi-Turn Spoken Language Understanding.. In *INTERSPEECH*. 3245–3249.

[4] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190* (2018).

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. 4171–4186.

[6] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. *arXiv preprint arXiv:1905.03197* (2019).

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[8] 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-Volume 70*. JMLR. org, 1587–1596.

[9] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised Learning with Deep Generative Models. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3581–3589. http://papers.nips.cc/paper/5352-semi-supervised-learning-with-deep-generative-models.pdf

[10] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[11] Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S. Yu. 2021. Augmenting Sequential Recommendation with Pseudo-PriorItems via Reversely Pre-training Transformer. *Proceedings of the 44th international ACM SIGIR conference on Research and development in information retrieval*.

[12] Nikolaos Malandrakis, Minmin Shen, Anuj Goyal, Shuyang Gao, Abhishek Sethi, and Angeliki Metallinou. 2019. Controlled Text Generation for Data Augmentation in Intelligent Artificial Agents. *arXiv preprint arXiv:1910.03487* (2019).

[13] Hoang Nguyen, Chenwei Zhang, Congying Xia, and S Yu Philip. 2020. Semantic Matching and Aggregation Network for Few-shot Intent Detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. 1209–1218.

[14] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*. 4077–4087.

[15] 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*. 5998–6008.

[16] Jason W Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196* (2019).

[17] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* (2016).

[18] Congying Xia, Caiming Xiong, S Yu Philip, and Richard Socher. 2020. Composed Variational Natural Language Generation for Few-shot Intents. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. 3379–3388.

[19] Congying Xia, Wenpeng Yin, Yihao Feng, and Philip Yu. 2021. Incremental Few-shot Text Classification with Multi-round New Classes: Formulation, Dataset and System. *arXiv preprint arXiv:2104.11882* (2021).

[20] Congying Xia, Chenwei Zhang, Hoang Nguyen, Jiawei Zhang, and Philip Yu. 2020. CG-BERT: Conditional Text Generation with BERT for Generalized Few-shot Intent Detection. *arXiv preprint arXiv:2004.01881* (2020).

[21] Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and S Yu Philip. 2018. Zero-shot User Intent Detection via Capsule Neural Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3090–3099.

[22] Congying Xia, Chenwei Zhang, Jiawei Zhang, Tingting Liang, Hao Peng, and Philip S. Yu. 2020. Low-shot Learning in Natural Language Processing. In *2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI)*. 185–189. https://doi.org/10.1109/CogMI50398.2020.00031

[23] Pawel Swietojanski Xingkun Liu, Arash Eshghi and Verena Rieser. 2019. Benchmarking Natural Language Understanding Services for building Conversational Agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*. Springer, Ortigia, Siracusa (SR), Italy, xxx–xxx. http://www.xx.xx/xx/

[24] Hu Xu, Bing Liu, Lei Shu, and P Yu. 2019. Open-world Learning and Application to Product Classification. In *The World Wide Web Conference*. ACM, 3413–3419.

[25] Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *ASRU*. 78–83.

[26] Kang Min Yoo, Youhyun Shin, and Sang-goo Lee. 2019. Data augmentation for spoken language understanding via joint variational generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 7402–7409.

[27] Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, S Yu Philip, Richard Socher, and Caiming Xiong. 2020. Discriminative Nearest Neighbor Few-Shot Intent Detection by Transferring Natural Language Inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 5064–5082.

[28] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 654–664.