PROTODA: EFFICIENT TRANSFER LEARNING FOR FEW-SHOT INTENT CLASSIFICATION

Manoj Kumar[1]*, Varun Kumar*[2]*, Hadrien Glaude*[2]*, Cyprien de Lichy*[2]*, Aman Alok*[2]*, Rahul Gupta*[2]

[1] Signal Analysis and Interpretation Lab, USC, Los Angeles, CA
[2] Amazon Alexa, Cambridge, MA

ABSTRACT

Practical sequence classification tasks in natural language processing often suffer from low training data availability for target classes. Recent works towards mitigating this problem have focused on transfer learning using embeddings pre-trained on often unrelated tasks, for instance, language modeling. We adopt an alternative approach by transfer learning on an ensemble of related tasks using prototypical networks under the meta-learning paradigm. Using intent classification as a case study, we demonstrate that increasing variability in training tasks can significantly improve classification performance. Further, we apply data augmentation in conjunction with meta-learning to reduce sampling bias. We make use of a conditional generator for data augmentation that is trained directly using the meta-learning objective and simultaneously with prototypical networks, hence ensuring that data augmentation is customized to the task. We explore augmentation in the sentence embedding space as well as prototypical embedding space. Combining meta-learning with augmentation provides upto 6.49% and 8.53% relative F1-score improvements over the best performing systems in the 5-shot and 10-shot learning, respectively.

Index Terms— meta learning, prototypical networks, data hallucination

1. INTRODUCTION

Intent classification (IC) is an important natural language processing task of voice controlled intelligent agents such as Amazon Alexa, Google Home, and Apple Siri. One of the first steps in such applications after converting speech to text is intent classification, where user queries are tagged with a sequence-level label identifying the underlying intent. To increase the capabilities of such agents, new intents are frequently added to the existing collection. Often, the development of a new intent starts with a few examples since labeled training data is scarce and expensive to obtain. Intent classification, in this case, resembles a few-shot learning setting where the goal is to generalize from a handful of training samples.

A natural resource available during new intent development is the collection of intents in-use by the voice controlled agent. These intents are often drawn from multiple domains such as music, reservations, dining, etc. and represent significant content variability between them. While transfer learning from intents in-use tries to borrow high level feature representations during new intent development, it is prone to overfitting in the few-shot setting case. On the contrary, learning from a diverse set of intents falls under the purview of meta-learning [1, 2], which learns across a collection of tasks as opposed to traditional supervised learning which learns across samples. Further, meta-learning has shown success in few-shot learning in computer vision [2, 3] and NLP [4, 5, 6], which can be useful for learning from a few annotated samples.

Another issue that is associated with few-shot learning is its susceptibility to sampling bias, i.e., estimated class distributions may not resemble the true population distribution due to limited sample availability. A popular approach towards mitigating this bias is *learning* to artificially synthesize new samples, known as data augmentation (DA). DA has been an active research topic in NLP over the years. A number of DA techniques have been experimented ranging from random perturbation [7] to deep-learning based approaches such as variability mode transfer between classes [8, 9]. However, most of them perform augmentation independent of the task, i.e distribution of generated samples is independent of task loss.

In this work, we optimize data augmentation model using task-specific objective, and combine it with meta learning to improve intent classification performance in the few-shot setting. In particular, we propose *ProtoDA* which jointly trains a conditional generator network [10] with prototypical networks [11, 12] (ProtoNets) to generate task specific samples. Combining data augmentation and ProtoNets is particularly suited, since during the few-shot setting prototypes are computed using a very limited number of examples, which incurs a sampling bias. Instead, computing prototype using both real and synthetic embedded examples allows to better estimate the class-specific population mean of the training set distribution for the class. We show the effectiveness of our method on intent classification task using open source datasets and a pro-

duction scale corpora. Our primary contributions are: 1) combining meta-learning and data augmentation as an alternative to conventional transfer-learning specifically for low-resource IC, and 2) introducing data augmentation in the ProtoNet embedding space for improving task performance in NLP.

2. RELATED WORK

2.1. Meta-learning using Prototypical Networks

Meta-learning, or learning-to-learn, is a learning paradigm that learns at two-levels: within a task; and across multiple tasks while leveraging common knowledge among them. The accumulated knowledge from an ensemble of tasks is used to improve few-shot accuracy on the target task, often on unseen classes. Various meta-learning approaches have been proposed mainly in the field of computer vision [2, 3].

ProtoNets [11] were first proposed for few-shot image classification with a relatively simple inductive bias when compared to other metric-learning methods such as matching networks [13] and relation networks [14]. ProtoNets are trained in an episodic manner using multiple tasks, with both tasks and train-test splits sampled within each episode (an episode refers to a single backpropagation step during the training process). Few approaches exist which apply ProtoNets in NLP. In [4], ProtoNets were trained using a weighted sum of metrics to handle diverse tasks. In [5], the authors use an attention mechanism to weigh both features and samples during distance computation in the embedding space. In this work, we use the original formulation of ProtoNets and propose a joint data augmentation for the few-shot learning task.

2.2. Data Augmentation with Meta-Learning

DA techniques in NLP have been explored on the lexical space including synonym replacement [15], back-translation [16] and sentence-level augmentation by replacing words with outputs from a language model [17]. Feature-space augmentation techniques on the other hand, generate new samples at the embedding space which are added to real samples during model training. Applications for DA include natural language generation [18], visual question answering [19], relation classification [20] and machine translation [21]. Most of the previous works focus on first training a data augmentation model, followed by adhoc data generation to augment the training set for the final task. Recently, [22] proposed an end-to-end data *hallucination* method that is trained using classification (task) loss. The hallucinator, a conditional generator, takes as inputs an original sample from the low-resource task and a noise sample, and generates a perturbed version of the samples which are augmented with the original training set. Inspired by this work, we explore a joint training of generator and classifier models in a meta-learning setup. The classifier (protonet in our case) is trained on the augmented set, including real as well as hallucinated samples. On the other hand, gradients from the classifier are propagated to the hallucinator for weight update. The feature extractor weights are typically pretrained and frozen while training the hallucinator. This setup encourages the hallucinator to generate samples that improve task performance and does not necessarily prioritize generating realistic-looking data samples.

3. MODELING SETUP

Our modeling setup involves learning latent representations of text (e.g. sentence embeddings) using an encoder network, followed by classification using ProtoNets. We consider adding the hallucinator at two separate points during training. We describe the setup of these components and the modeling architecture below.

3.1. Encoder Network

Following [15, 23, 24], we use a neural network encoder in all our experiments to extract sentence-level embeddings from text. The encoder takes in a combination of character-level and word-level representations at the input. Character representations are learnt using a 2-D convolution neural network. One-hot character encodings (of dimension 32) are passed through 2 convolutional layers (with a kernel size of 5) with max pooling followed by a temporal pooling layer to form a word-level representation. A dropout with a probability of 0.2 is used in both layers for regularization. The CNN output is concatenated with pre-trained word-level embeddings (GloVe [25]; 100-dimensional). The resulting word-level representation is then passed through a Bi-LSTM with a 128-dimensional hidden state to obtain contextualized word embeddings. A statistics-pooling layer computes the minimum, maximum and mean values of these embeddings to obtain sentence-level embeddings.

3.2. Prototypical Networks

ProtoNets learn a non-linear transformation where each class is reduced to a single point, specifically the centroid (prototype) of examples from that class. During inference a test sample is assigned to the class of nearest centroid. Following, we illustrate a single episode of ProtoNet training, then extend it to multiple training tasks.

3.2.1. Episodic training

Given a task t, consider a set of labeled training embeddings $D_t = (\mathbf{X_{train}}, \mathbf{Y_{train}}) = (\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), ...$ $(\mathbf{x_{Nsamples}}, y_{N\,samples})$ where $\mathbf{x_i} \in \mathbb{R}^M$ and $y_i \in \{1, 2, .., C\}$. D_t is sampled to form two sets: *supports* (S_t) are used for prototype computation while *queries* (Q_t) are used for estimating class posteriors and loss computation. S_t and Q_t are

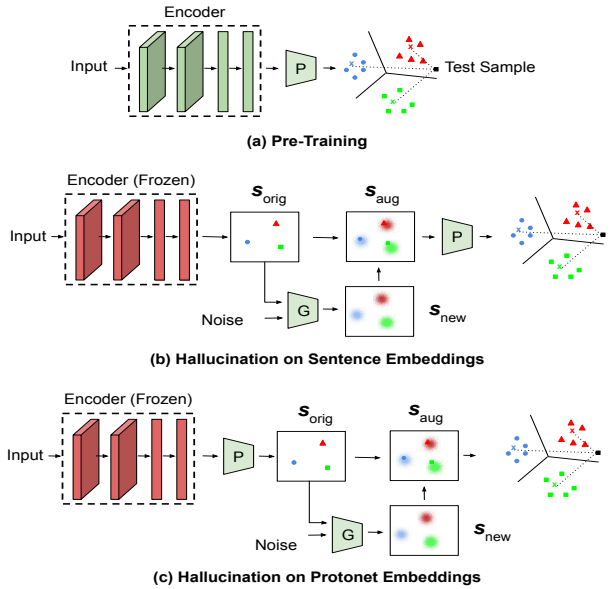

Fig. 1. Data augmentation in ProtoNets. The encoder is pre-trained using prototypical loss and it's weights frozen during hallucination. The hallucinator (G, a conditional generator) is trained with the ProtoNet's (P) loss function.

not necessarily mutually exclusive. ProtoNets learn a mapping $f_\theta : \mathbb{R}^M \rightarrow \mathbb{R}^P$ where the prototype of each class is computed as follows:

$$v_c = \frac{1}{|S_{t,c}|} \sum_{(\mathbf{x_i}, y_i) \epsilon S_{t,c}} f_\theta(\mathbf{x_i}) \tag{1}$$

$S_{t,c}$ is the set of all examples in S_t belonging to class c. For every test sample $\mathbf{x} \in Q_t$, the posterior probability given class c is as follows:

$$p(y = c|\mathbf{x}) = \frac{\exp\left(-d\left(f_\theta(\mathbf{x}), \mathbf{v}_c\right)\right)}{\sum_{c' \in C} \exp\left(-d\left(f_\theta(\mathbf{x}), \mathbf{v}_{c'}\right)\right)} \tag{2}$$

d represents the distance function. Euclidean distance was chosen based on empirical results in the original ProtoNet implementation [11]. Learning proceeds by minimizing the negative log probability for the true class using gradient descent. Loss for the episode is computed as follows:

$$L = -\frac{1}{|Q_t|} \sum_{(\mathbf{x_i}, y_i) \epsilon Q_t} \log(p_\theta(y_i = c \mid \mathbf{x_i})) \tag{3}$$

In general, a different task is chosen for each episode. ProtoNets, and meta-learning in general benefit from a large number of training tasks. In this work, we treat each training corpus as a task, and select all classes from the task within an episode. Pseudocode for ProtoNet training is provided in Algorithm 1.

Algorithm 1 Extending episodic learning to multiple tasks in the training corpus. SAMPLE(S, K) denotes selecting K samples uniformly at random from set S with replacement.

Input: T: set of tasks, N_{tasks}: number of episodes

for i $\in \{1...N_{tasks}\}$ **do**
  $t \leftarrow$SAMPLE$(T, 1)$. ▷ Sample a task
  for c in $\{1...C\}$ **do**
    $D_{t,c} \leftarrow$ Embeddings $\in$ class c in task t
    $S_{t,c} \leftarrow$SAMPLE$(D_{t,c}, k)$ ▷ k supports
    $Q_{t,c} \leftarrow$SAMPLE$(D_{t,c}, q)$ ▷ q queries
  Perform Episodic Training: Equations (1-3)

The ProtoNet model architecture in this work consists of two feed-forward layers with 128 units in each layer. Hence, the model takes as input 768-dimensional embeddings from the sentence encoder and outputs 128-dimensional ProtoNet embeddings. Similar to the sentence encoder, dropout with a probability of 0.2 is used for regularization in both layers.

At each episode, we sample k supports (the value of k is experimented with 5 and 10) and 10 queries per class. The number of classes C varies according to the task (i.e training corpus). The entire network is trained with Adam optimizer (lr=0.001, β_1=0.9, β_2=0.99) using the PyTorch toolkit.

3.3. Hallucination

During meta-training, the learning setting (*N-way, k-shot*: N classes, k samples/class) is carefully controlled to resemble the testing scenario. In [11] for instance, the authors matched the k-shot setting during both meta-training and meta-testing. While additional examples can be sampled from the class to reduce sampling bias during meta-training, data augmentation can introduce additional variations that are otherwise not present in the original data. In this work, we tie the augmentation process directly with the task objective (i.e intent classification). At every episode, gradients computed using the task loss are used to update not just the ProtoNet, but also the conditional generators used for data augmentation (hallucination).

Let $S_{c,orig}$ represent supports from class c during an episode of meta-training. Let $S_{c,new}$ represent a subset of $S_{c,orig}$ chosen for augmentation. Each sample $\in S_{c,new}$ is passed as input to a generator network (G) alongwith a noise vector to produce an augmented sample. The new prototype for class c is computed as centroid of $S_{c,aug} = S_{c,orig} \cup S_{c,new}$:

$$v_{c,aug} = \frac{1}{|S_{c,aug}|} \left(\sum_{\mathbf{e}_i \in S_{c,orig}} f_\theta(\mathbf{e}_i) + \sum_{\mathbf{e}_j \in S_{c,new}} f_\theta(G(\mathbf{e}_j, \mathbf{z})) \right) \tag{4}$$

where $\mathbf{e} = E(\mathbf{x})$, E represents the sentence encoder described in Section 3.1 and $\mathbf{z}$ is a noise vector with same di-

mensionality as $\mathbf{e}$. The generator training does not have a separate objective of its own, but updates according to the episodic loss in Equation (3). While the method described above augments samples in the sentence embedding space (Figure 1b), an alternative approach is to augment samples in the ProtoNet embedding space (Figure 1c), i.e

$$v_{c,aug} = \frac{1}{|S_{c,aug}|}\left(\sum_{\mathbf{e}_i \in S_{c,orig}} f_\theta(\mathbf{e}_i) + \sum_{\mathbf{e}_j \in S_{c,new}} G(f_\theta(\mathbf{e}_j), \mathbf{z})\right) \quad (5)$$

We meta-train the hallucinator network as follows: First, the sentence encoder and ProtoNet are pre-trained for 20000 episodes. Next, the sentence encoder weights are frozen, while the generator network (two feed-forward layers with 128 units in each layer, dropout with a probability of 0.2) and ProtoNet are trained together for another 20000 episodes. Following [22], the hallucinator weights are initialized with block diagonal identity matrices. At each episode, 20% of the original samples are randomly selected for hallucination, hence $|S_{c,aug}| = 1.2 \times |S_{c,orig}|$. The generator weights are frozen during meta-testing, and used to augment the supports for prototype computation.

4. DATASETS

We use two source corpora: the task-oriented dialog corpus from Facebook (FB) [26] containing crowd-sourced annotations for queries from the navigation and event management domains, and the Air Travel Information System (ATIS) corpus [27] consisting of spoken queries from the air travel domain. Both corpora contain natural language queries (as opposed to written form) and are more suitable to our target domain, i.e voice-controlled agents. We remove intents with less than 20 utterances from both corpora and utterances with multiple root intents from FB. This results in a total of 45,489 utterances from 25 intents.

For evaluation purpose, we use the SNIPS corpus [28] which has served as a benchmark for recent sequence classification tasks in NLP. SNIPS contains crowd-sourced queries from seven intents with a balanced sample distribution across classes: $\approx$ 2000 samples for training and 100 samples for validation for each intent. We divide the seven intents in SNIPS into train (BookRestaurant, AddToPlaylist, RateBook, SearchScreeningEvent) and test (PlayMusic, GetWeather, SearchCreativeWork) to evaluate within different experimental configurations (see Section 5.1)

Similar to previous meta-learning works in computer vision [2, 11] which have used hundreds of classes during meta-training, we curate an Alexa corpus to aid the unseen intent configuration (Section 5.1). The Alexa corpus consists of user queries directed at the devices supported by the smart agent. Voice queries were manually transcribed and labeled for intents. The queries span 68 Alexa third-party *skills*[1] and $\approx$1100 intents. The number of intents per skill ranges between 2 to 30. Treating each Alexa skill as a task (Section 3.2.1), the augmented training corpus containing FB, ATIS, SNIPS and Alexa contains 71 training tasks.

Table 1. Training intents used in each experimental setup. SNIPS-4 refers to BookRestaurant, AddToPlaylist, RateBook, SearchScreeningEvent classes.

	Seen Intents	Unseen Intents
Single Task	SNIPS (All)	SNIPS-4
Multi Task	FB,ATIS + SNIPS (All)	FB,ATIS + SNIPS-4

5. EXPERIMENTS

5.1. Protonets for Transfer Learning

In the first set of experiments, we evaluate protonets for transfer learning under two conditions: seen intents and unseen intents. For seen intents, we make use of the train partitions from test intents for model backpropagation. This may not be always possible, when for instance, on-device computation for model adaptation is restricted/infeasible. Nevertheless, these experiments analyze the value of including related tasks during ProtoNet training. We repeat the experiments by removing test intents during training time, which more accurately represents the scenario where we wish to evaluate pre-trained models on newly-introduced skills (intents) for a voice-controlled assistant. For the case of seen intents, we develop a competitive conventional transfer learning method (Conv TL) to compare with ProtoNets - We use the sentence encoder described in Section 3.1 and add two feed-forward layers (128 units in each layer, dropout with probability of 0.2) similar to the ProtoNet architecture. The model is trained to minimize cross-entropy loss on FB, ATIS and SNIPS (train intents). Following, the training partition from the test intents are used to fine-tune the network by replacing the final softmax layer.

For both seen and unseen intents, we experiment with two different setups by varying the number of tasks available during meta-training. Under the single-task setup, we use only the SNIPS corpus during meta-training. Here, transfer learning happens in the case of unseen intents. Under the multi-task setup we make use of 3 tasks - FB, ATIS and SNIPS corpora. Increasing the number of tasks is expected to improve the learning capability of ProtoNets. Table 1 illustrates the proposed experiment setups.

[1] https://developer.amazon.com/en-US/alexa/alexa-skills-kit

Table 4. Micro-F1 scores (%) using different augmentation methods (None, Noise: standard normal, Hall: Hallucination) and embeddings (Sent: Sentence, Proto: Protonet) for transfer learning on seen and unseen intents. ± indicates 95% confidence interval.

Augmentation	Seen (5-shot)		Seen (10-shot)		Unseen (5-shot)		Unseen (10-shot)	
	Sent	Proto	Sent	Proto	Sent	Proto	Sent	Proto
None	75.60 ± 4.27		86.40 ± 1.91		79.85 ± 1.43		89.02 ± 1.24	
Noise	75.72 ± 3.63	**77.47 ± 3.66**	86.85 ± 1.95	86.93 ± 1.93	80.57 ± 1.74	82.17 ± 1.42	89.87 ± 1.00	89.93 ± 0.79
Hall	**76.30 ± 3.10**	76.62 ± 3.52	**87.11 ± 1.88**	**88.08 ± 1.88**	**81.33 ± 1.87**	**83.67 ± 1.65**	**90.38 ± 0.87**	**91.18 ± 0.73**

5.2. Data Augmentation

In the next set of experiments, we select the best performing configurations from seen and unseen intents cases and perform data hallucination (Hall) during training. As mentioned in Section 3.3, we train the hallucinator at one of two spaces, sentence embeddings or ProtoNet embeddings. At each space, we compare hallucination with random perturbation data augmentation (Noise) which has been shown as a competitive baseline [29] in few-shot data augmentation experiments. Moreover, we control the amount of random perturbation per class to match that of hallucination (20% i.e., we introduce one synthetic embedding for every 5 real embeddings) thereby creating a fair comparison with hallucination. In this method, we augment the embeddings with additive and multiplicative noise generated using a normal distribution with zero-mean and standard variance of 10% batch variance in every dimension.

During evaluation, all experiments including the baseline are repeated for 20 trials by randomly selecting k (= 5,10) labeled examples from the train partitions for prototype computation. The validation partitions from test intents are used for evaluation. We report the averaged micro F1 scores alongwith the 95% confidence intervals for all experiments.

Table 2. Micro-F1 scores (%) for TL with seen intents

Method	5-shot	10-shot
Conv TL	74.98 ± 3.46	82.02 ± 3.94
Single Task	70.48 ± 4.20	82.48 ± 3.27
Multi Task	**75.60 ± 4.27**	**86.40 ± 1.91**

Table 3. Micro-F1 scores (%) for TL with unseen intents

Method	5-shot	10-shot
Single Task	49.95 ± 4.79	57.45 ± 3.33
Multi Task	70.82 ± 0.97	73.18 ± 0.54
Across-Domain + Alexa	**79.85 ± 1.43**	**89.02 ± 1.24**

6. RESULTS AND DISCUSSION

Tables 2 and 3 show the performance on seen intents and unseen intents respectively. In the former, we observe that single task transfer does not provide significant gains over ConvTL, even failing to outperform in the 5-shot case. We observe that ProtoNets benefit with increased task variability during multiple tasks, where transfer learning happens across corpora. While unseen intents in general result in lesser classification performance owing to non-availability of classes during training, gains from increased task variability (during multiple tasks) are significant. Specifically, the 5-shot and 10-shot settings result in 20.87% and 15.73% absolute improvement over the single-task, in comparison to 5.12% and 3.92% during seen intents. When the number of training tasks is greatly increased using the Alexa corpus, the gains in classification outperform the best performing models in seen intents, including ConvTL. These results demonstrate the importance of variability in training tasks for meta-learning.

In Table 4, we see that both augmentation methods improve performance across the different settings. In most cases, hallucination improves over random perturbations, as it can learn to de-bias prototypes computed from a very small set of examples. Within each TL method and k-shot setting, ProtoNet embeddings prove to be a better choice for DA over sentence embeddings. We believe that the smaller dimensionality in the ProtoNet space and proximity to the training objective function (ProtoNet loss) makes hallucination more effective in the protonet embedding space.

7. CONCLUSION

Conventional TL approaches for low-resource NLU applications are still dependent on a small number of labeled samples from unseen intents during model training. In this work, we propose an alternative approach by combining meta-learning with data hallucination. Given sufficient variability in the training set (represented as tasks), we show that ProtoNets outperform models trained with standard cross-entropy objectives. Data augmentation further assists generalization by reducing sampling bias during prototype computation. While augmenting samples with additive and multiplicative noise is beneficial, we show better improvements by learning to optimize the hallucinator directly with the task loss. In the future, we would like to extend this approach to downstream NLU tasks for voice controlled agents, such as named entity recognition which entails sequence labels.

8. REFERENCES

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas, "Learning to learn by gradient descent by gradient descent," in *Advances in neural information processing systems*, 2016, pp. 3981–3989.

[2] Sachin Ravi and Hugo Larochelle, "Optimization as a model for few-shot learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1126–1135.

[4] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou, "Diverse few-shot text classification with multiple metrics," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 2018, pp. 1206–1215.

[5] Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun, "Hybrid attention-based prototypical networks for noisy few-shot relation classification," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, Jan 2019, pp. 6407–6414.

[6] Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He, "Natural language to structured query generation via meta-learning," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, June 2018, pp. 732–738, Association for Computational Linguistics.

[7] Terrance DeVries and Graham W Taylor, "Dataset augmentation in feature space," *arXiv preprint arXiv:1702.05538*, 2017.

[8] Bharath Hariharan and Ross Girshick, "Low-shot visual recognition by shrinking and hallucinating features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3018–3027.

[9] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein, "Delta-encoder: an effective sample synthesis method for few-shot object recognition," in *Advances in Neural Information Processing Systems*, 2018, pp. 2845–2855.

[10] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[11] Jake Snell, Kevin Swersky, and Richard Zemel, "Prototypical networks for few-shot learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 4080–4090.

[12] Sai Kumar Dwivedi, Vikram Gupta, Rahul Mitra, Shuaib Ahmed, and Arjun Jain, "Protogan: Towards few shot learning for action recognition," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[13] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al., "Matching networks for one shot learning," in *Advances in neural information processing systems*, 2016, pp. 3630–3638.

[14] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1199–1208.

[15] Xiang Zhang, Junbo Zhao, and Yann LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[16] Rico Sennrich, Barry Haddow, and Alexandra Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Aug. 2016, pp. 86–96, Association for Computational Linguistics.

[17] Sosuke Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 452–457.

[18] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing, "Toward controlled generation of text," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1587–1596.

[19] Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan, "Data augmentation for visual question answering," in *Proceedings of the 10th International Conference on Natural Language Generation*, 2017, pp. 198–202.

[20] Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin, "Improved relation classification by deep recurrent neural networks with data augmentation," *arXiv preprint arXiv:1601.03651*, 2016.

[21] Marzieh Fadaee, Arianna Bisazza, and Christof Monz, "Data augmentation for low-resource neural machine translation," *arXiv preprint arXiv:1705.00440*, 2017.

[22] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan, "Low-shot learning from imaginary data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7278–7286.

[23] Jason PC Chiu and Eric Nichols, "Named entity recognition with bidirectional lstm-cnns," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 357–370, 2016.

[24] Dongyun Liang, Weiran Xu, and Yinge Zhao, "Combining word-level and character-level representations for relation classification of informal text," in *Proceedings of the 2nd Workshop on Representation Learning for NLP*, 2017, pp. 43–47.

[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[26] Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis, "Semantic parsing for task oriented dialog using hierarchical representations," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018, pp. 2787–2792.

[27] Charles Hemphill, John Godfrey, and George Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania*, 1990.

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

[29] Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell, "A closer look at feature space data augmentation for few-shot intent classification," in *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, Hong Kong, China, Nov. 2019, pp. 1–10, Association for Computational Linguistics.