CosPlace Extended: Insights into Architectural and Feature Choices for Visual Geo-localization

Giulia Chillemi
s296737@studenti.polito.it

Valeria Bo
s304645@studenti.polito.it

Alessandra Borzomì
s302523@studenti.polito.it

Abstract

In this paper, we extend the CosPlace framework for Visual Geo-localization (VG), allowing for the construction, training, and evaluation of various architectures. Our objective is to gain insights into the effectiveness of specific features and design choices in a VG pipeline, establishing a systematic evaluation protocol for method comparison. Leveraging our framework, we experiment extensively to benchmark and optimize parameters, while evaluating the impact of engineering techniques on model performance. The entire codebase is available at https://github.com/glichill/CosPlace.

1. Introduction

Visual Geo-localization has emerged as a prominent task in various domains. These include augmented reality, navigation systems, and urban planning, all of which aim to estimate the location of an image or a scene. Deep learning approaches can treat it as an image retrieval problem, comparing query images to geo-tagged databases. Neural networks project images into an embedding space that captures location similarity, enabling image retrieval. The main challenges of this approach are limited representative datasets and scalability issues.
The CosPlace model [5] addresses these by introducing two main novelties: the San-Francisco XL dataset and a novel training procedure. The first surpasses previous datasets in size and data sparsity while the latter uses a classification task as a proxy, extracting discriminative descriptors for retrieval.
In this paper, we aim to enhance performance by exploring the following extensions to the model:

1. **Data augmentation techniques:** By augmenting the training data with various augmentation techniques we aim to enhance the model's ability to handle variations in lighting conditions, viewpoints, and environmental factors.
2. **Alternative approaches to the final aggregation stage:** By investigating different aggregation strategies, we aim to improve the model's ability to capture and integrate spatial information effectively.
3. **Domain adaptation techniques:** By adapting the model to different target domains, we expect to improve its performance and robustness in real-world scenarios.
4. **New optimizers:** By experimenting with advanced optimization techniques, such as adaptive learning rate methods or optimizer variants, we aim to improve the convergence speed and overall performance of the model.

2. Related Work

2.1. Data Augmentation

Data augmentation plays a crucial role in image retrieval by enhancing the performance and robustness of the retrieval system. Augmentation techniques enable the system to handle variations in visual appearance, increase robustness to noise and imperfections, improve generalization to unseen data, and effectively capture the underlying features that contribute to image similarity. The underlying assumption is that by introducing augmentations, more valuable information can be extracted from the original dataset [11]. This assumption has been supported by studies showcasing improved general performance of deep learning networks [14], leading to a reduction in overfitting [1].

2.2. Aggregation

1. **NetVLAD:** NetVLAD is a specialized deep learning model for place recognition, a task vital in robotics, mapping, and autonomous navigation. It builds on the VLAD (Vector of Locally Aggregated Descriptors) method [4] [3], which converts local image descriptors into a global representation. NetVLAD enhances VLAD by using neural networks to adaptively learn both the descriptors and the cluster centers. This allows it to manage variations in appearance and view-

point effectively. Its top-tier performance in various benchmarks highlights its utility in computer vision applications for place recognition.

2. **MixVPR:** MixVPR [2] is an innovative approach to Visual Place Recognition (VPR), designed to address the challenges posed by large-scale environments, varying weather conditions, and illumination changes. Unlike traditional methods such as NetVLAD, which rely on local or pyramidal aggregation of features, MixVPR introduces a holistic feature aggregation technique called Feature-Mixer. Feature maps extracted from pre-trained CNN backbones serve as a set of global features, and Feature-Mixer incorporates global relationships between these maps in a cascading manner. The architecture entirely consists of multilayer perceptrons (MLPs), making it an all-MLP design. MixVPR has demonstrated state-of-the-art performance on multiple large-scale benchmarks [2], outperforming existing techniques like CosPlace [5] and NetVLAD with fewer parameters. Furthermore, it has shown to be orders of magnitude faster than other two-stages retrieval techniques.

2.3. Domain Adaptation

Adversarial Domain Adaptation (ADA) has emerged as an influential technique to support the robustness of place recognition models, particularly in diverse environmental conditions. ADA employs a dual architecture comprising a feature extractor and a domain discriminator, inspired by the original concept of Generative Adversarial Networks (GANs) [8]. The feature extractor is responsible for generating image descriptors, akin to the role of traditional ConvNet features in place recognition. On the other hand, the domain discriminator aims to classify these descriptors based on their originating domains. This setup facilitates a form of adversarial learning, reminiscent of the Domain-Adversarial Training of Neural Networks (DANN) [7]. The objective is to produce image descriptors that are domain-invariant, effectively mitigating the impact of domain shift on the model's performance [12]. Such an approach has shown promise in enhancing place recognition capabilities, especially when the model has to adapt to varying scenarios [3].

2.4. Optimizers and Schedulers

The role of the Optimizers is to adjust the parameters of a machine learning model during the training phase in order to minimize the loss function that measure the difference between the model's prediction and the ground truth. Some of the most popular optimizers are Stochastic Gradient Descent (SGD) and Adaptive Moment Estimation (Adam) with their variants AdamW and Avarage Stochastic Gradient Descent (ASGD).
The main difference between Adam and AdamW is that Adam [6] adjust the learning rates for each parameter individually applying weight decay to both the model's parameters and the moving averages of past squared gradients while AdamW [9] decouples weight decay applying it only to the model's parameters. For this reason AdamW avoids overly penalizing the learned weights obtaining a more stable training and improving generalization [10].
While ASGD [13] is a variant of the standard Stochastic Gradient Descent (SGD) algorithm because it uses the average of the gradients accumulated over a number of iterations for the update. Hence the ASGD algorithm take into account the history of the optimization process reducing the noise and the variance in the updates leading to more stable optimization performance.
Sometimes it might be usefull to optimize the optimization problem itself using the Schedulers that allow more effective exploration of the loss landscape, fine-tuning the learning rate and avoiding overshooting or getting stuck in local minima but this will not be part of this paper due to the low number of epoch used to run the experiments.

3. Methodology

Prior to investigate our experimental findings, it is necessary to conduct a comprehensive analysis of the datasets utilized in our research, namely, the *SF_XS* and *Tokyo_XS* datasets. A discernible distinction emerges when comparing the performance outcomes on the *SF_XS* validation set to those of the *SF_XS* test set, as visually depicted in Table 1.

	SF_XS val	SF_XS test	$Tokyo\_XS$
R@1	52.3	16.3	28.9
R@5	66.1	28.1	46.0

Table 1. Datasets Recalls

Figure 1. *SF_XS* (first row), *Tokyo_XS* (second row)

One of the factors contributing to the variance in performance results between the *SF_XS* validation and *SF_XS* test

(a) Starting figure (b) Random Horizontal Flip (c) Gaussian Blur (d) Color Jitter

Figure 2. Augmentation techniques

sets pertains to the inherent data distribution discrepancy. Specifically, the validation set comprises 7,993 queries and 8,015 database images, whereas the test set contains 1,000 queries and 27,191 database images.
It can also be noticed that the algorithm exhibits better performance in the *Tokyo_XS* test set despite the images in the queries show lower luminosity when compare to those in the *SF_XS* training set, see Figure 1. This observed phenomenon can be attributed to the algorithm's heightened sensitivity towards darker regions or patterns during the feature extraction process.

3.1. Data Augmentation

To address the considerations mentioned in the Related Work section, we implemented the following data augmentation techniques represented in Figure 2:

- *Orizontal flipping*: by horizontally flipping images along the vertical axis, this technique mirrors their content. It increases the model's robustness to variations in horizontal image orientation, making it adept at handling horizontally flipped or differently oriented query images.
- *Blurring*: blurring involves applying filters to simulate different levels of image blurriness. By reducing the impact of small details and imperfections, blurring enables the model to focus on more global features.
- *Color jittering*: introduces random color transformations such as adjusting brightness, contrast, saturation, and hue, incorporating variations in lighting conditions and color distributions.

3.2. Aggregation

1. **NetVLAD:** In our architecture, we employ a NetVLAD layer, a differentiable variant of the VLAD (Vector of Locally Aggregated Descriptors) algorithm, for place recognition. The NetVLAD layer is initialized using an unsupervised learning approach to set the cluster centroids. Specifically, we extract 50,000 descriptors from a random subset of the training dataset. These descriptors are normalized and fed into a k-means clustering algorithm. To determine the optimal number of clusters k, we run the k-means algorithm for varying sizes of k and compute the sum of squared distances (SSD) for each. An elbow graph is plotted in Figure 3 with these SSD values against their corresponding k values to identify the optimal number of clusters that minimizes intra-cluster variance while avoiding overfitting. Through this method, we empirically found that 15 is the optimal number of clusters. The centroids obtained from this k-means clustering are then used to initialize the NetVLAD layer's parameters.
This initialization strategy serves as a powerful unsupervised pre-training step, enhancing the layer's capability to generate discriminative and robust image descriptors for place recognition.

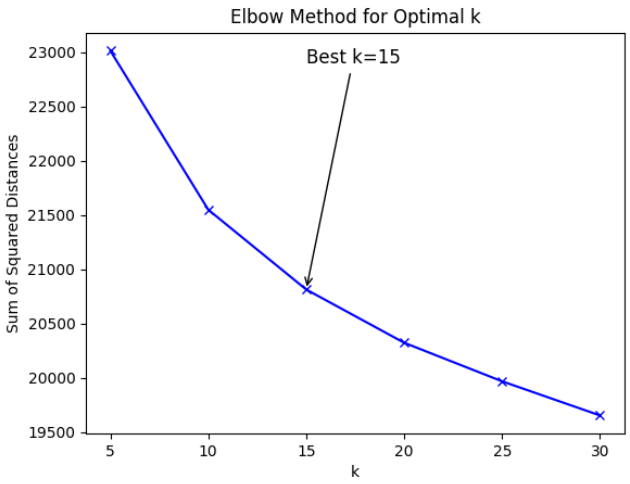

Figure 3. Elbow Graph

2. **MixVPR:** In our approach, we employ a novel vision-perceiver architecture that blends the strengths of convolutional networks with multi-layer perceptrons (MLPs) for efficient visual feature representation and retrieval. The architecture is inspired by the Vision Perceiver model presented in [2] , but with key customizations to suit our specific requirements. Our implementation consists of multiple stacked layers, termed as "Feature Mixer Layers," designed to blend spatial and channel-wise features. Each Feature Mixer Layer starts with Layer Normalization, followed by a sequence of linear transformations and activation functions. The depth of the stacked Feature Mixer Layers is set to 4 in our experiments. These layers are preceded by a channel projection layer and succeeded by a row projection layer, both realized through linear transformations. The output features are then normalized for the retrieval task. Our modified architecture accommodates an input feature map of dimensions $256 \times 20 \times 20$ and outputs a normalized vector with a size determined by the row-wise projection.

3.3. Domain Adaptation

3.3.1 St Lucia Dataset

The UQ St Lucia Dataset, Figure 4, was collected on a 9.5-km circuit at the University of Queensland's St Lucia campus on December 15, 2010. It comprises visual data from a calibrated stereo camera along with orientation and translation ground truth. The dataset includes a variety of driving conditions such as roadworks, speed bumps, and differing light environments.

Figure 4. St. Lucia Dataset

3.3.2 Adversarial Domain Adaptation (ADA)

To address the domain shift problem and improve the model's ability to generalize across different domains, we incorporate Adversarial Domain Adaptation (ADA) into our training pipeline. For experimentation, we use the *SF_XS* train set as the source domain and perform tests on two target domains: the *Tokyo_XS* test database and the *St_Lucia* test database, independently. The objective is to identify the target domain that best generalizes the source domain model, validated on the *SF_XS* test set. During training, domain labels are assigned to both source and target data, which are then passed through a domain discriminator network. This network is trained adversarially to minimize the domain distribution discrepancy between the source and target domains. The domain discriminator consists of a fully connected neural network with ReLU activations and outputs a scalar value, which is then used to compute adversarial losses for both source and target domains. These losses are combined with the main task loss using a hyperparameter that controls the weight of the adversarial loss.
Our approach effectively encourages the learned feature representations to be domain-invariant, thereby enhancing the model's generalization performance.

3.4. Optimizers

During the training of a machine learning model the choice of the hyperparameters, such as the learning rate and the weight decay, is essential in order to achieve an efficient optimization.
For this reason we compared the Adam Optimizer used in the original work [5] with AdamW and ASGD varying the hyperparameters:

- **learning rate (lr):** scalar hyperparameter that determines the step size for parameter updates during training.
- **weight decay (wd):** regularization technique used to prevent overfitting that add a penalty term to the loss function to encourage smaller weights and a simpler model.
- **lambd:** It is the decay term for the past weights used in the average calculation in ASGD.
- **alpha (al):** It is the power value that is used to update the learning rate.

A larger learning rate leads to larger steps, which can result in faster convergence but may cause overshooting or instability while a smaller one results in smaller steps but may lead to slower convergence. An high value for weight decay can lead to over-regularization and under-fitting, while a low value can result in under-regularization and over-fitting. For this reasons we tryed different combination of hyperparameters for each optimizer that are listed in Table 2.
As shown in Figure 5 and Figure 6 the optimizer ASGD is the one with the worst performances in both datasets and that's because we only used 2000 iterations, with a number

Case Number	Optimizer	Parameters
1	Adam	learning rate = e^{-5}
2	Adam	learning rate = $5e^{-6}$
3	Adam	learning rate = e^{-4}
4	AdamW	learning rate = e^{-5} weight decay = 0.001
5	AdamW	learning rate = $5e^{-6}$ weight decay = 0.001
6	AdamW	learning rate = e^{-4} weight decay = 0.001
7	AdamW	learning rate = e^{-5} weight decay = 0.0001
8	AdamW	learning rate = e^{-5} weight decay = 0.01
9	ASGD	learning rate = 0.001 lambda = 0.0001 alpha = 0.75
10	ASGD	learning rate = 0.001 weight decay = 0.001 alpha = 0.75

Table 2. Optimizer

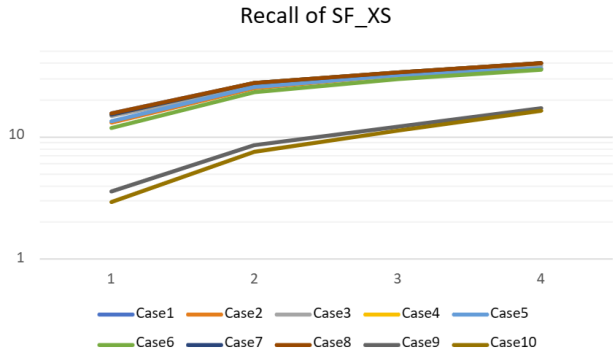

Figure 5

of iterations this small the optimizer struggles to accumulate meaningful information for the averaging of the gradient.

Meanwhile Adam and AdamW have almost the same trand with similar recall in the cases 1, 4, 7 and 8 for both the datasets.
For Adam and AdamW the worst results are obtained when we increase the learning rate because of the overshooting and also the results where we decrease the learning rate performs a little worse than the optimal one because a low value causes difficulty in convergence.

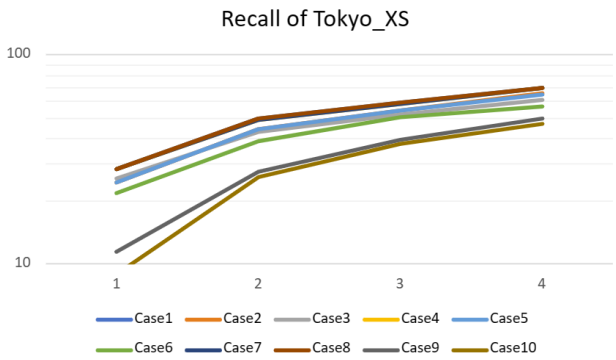

Figure 6

4. Experimental results

In this part of the report, the outputs of the models for different extensions that we implemented are available.

4.1. Data augmentation

Based on the insights gleaned from Table 3, it is evident that setting the contrast to a range of $[1.5, 2.0]$ outperforms the baseline model for the *SF_XS* dataset. However, this improvement is accompanied by a slight decrease in the performance for *Tokyo_XS*. Interestingly, the Gaussian blur technique does not contribute to any significant enhancement in the model's overall performance for either dataset. Conversely, employing the random horizontal flip technique yields a noticeable uptick in the recall rates R@1, R@5, R@10, and R@20 for the *Tokyo_XS* dataset.
Based on previous results with various data augmentation techniques, we have fine-tuned our model to incorporate a contrast range of $[1.5, 2.0]$, a random horizontal flip with a parameter value of 0.5, and a random resized crop factor of 0.5. With this optimized configuration set, we subsequently implemented the following techniques.

4.2. Aggregation Layer & Domain Adaptation

Both NetVLAD and MixVPR are effective feature aggregation methods, but our empirical evaluations point to a different hierarchy of performance. As indicated in Table 4, MixVPR consistently outperforms NetVLAD on both the *SanFrancisco_XS* and *Tokyo_XS* datasets. This suggests that MixVPR is not only computationally efficient but also more adept at handling the various challenges commonly associated with geo-localization tasks, including changes in viewpoint and lighting conditions.

Utilizing the Adversarial Domain Adaptation technique along with the NetVLAD aggregator, we observed a notable improvement in the model's performance on the *SF_XS* dataset (Table 5). This enhancement led to better model generalization when tested on *SF_XS*. Additionally, we eval-

	SF_XS				Tokyo_XS			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
Baseline	16.3	28.1	34.0	40.1	28.9	46.0	59.0	71.1
Random Horizontal Flip	15.1	27.1	32.6	37.9	27.6	51.7	61.9	**72.1**
Gaussian Blur (kernel_size=5, sigma=(0.5,1))	14.5	25.3	32.1	38.3	26.1	49.8	60.0	70.1
Color-Jitter with contrast [1.0, 1.5]	**19.7**	**33.0**	37.9	43.6	**37.8**	**53.7**	59.0	70.2
Color-Jitter with contrast [1.5, 2.0]	19.5	32.1	**38.3**	**43.8**	36.5	52.7	**62.2**	70.8
Color-Jitter with contrast [3.0, 4.0]	16.9	30.1	35.8	41.9	30.8	49.2	54.0	66.0

Table 3. Augmentation results

Aggregator	SF_XS				Tokyo_XS			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
GeM	19.1	30.4	36.2	43.1	38.1	55.9	63.5	71.4
NetVLAD	22.8	39.0	44.7	51.0	36.2	55.2	64.4	72.7
MixVPR	**30.9**	**44.4**	**51.8**	**57.4**	**48.9**	**68.9**	**75.9**	**81.6**

Table 4. Aggregator

Source Domain	Target Domain	SF_XS Test
SF_XS train	Tokyo_XS database	R@1: 24.7
		R@5: **39.0**
		R@10: **46.4**
		R@20: **54.0**
SF_XS train	St_Lucia database	R@1: **24.8**
		R@5: 38.3
		R@10: 44.2
		R@20: 51.3

Table 5. Domain Adaptation. Testing on SF_XS.

Method	St_Lucia	Tokyo_XS
Baseline Model	R@1: 88.7	R@1: 36.2
	R@5: 94.6	R@5: **55.2**
	R@10: 96.4	R@10: **64.4**
	R@20: 98.2	R@20: **72.7**
Baseline Model + Domain Adaptation	R@1: **92.1**	R@1: **37.5**
	R@5: **97.2**	R@5: 52.7
	R@10: **98.3**	R@10: 60.0
	R@20: **99.2**	R@20: 66.3

Table 6. Domain Adaptation. Testing on St_Lucia and Tokyo_XS.

uated the model on two target domains, using *St_Lucia* as the test set with *Tokyo_XS* as the target domain and vice versa, see Table 6. Specifically, the recall rates for *St_Lucia* exhibited a significant increase when *Tokyo_XS* was used for

Aggregator	Optimizer	SF_XS	Tokyo_XS
MixVPR	Adam	**R@1: 30.9**	R@1: 48.9
		R@5: 44.4	**R@5: 68.9**
		R@10: 51.8	R@10: 75.9
		R@20: 57.4	R@20: 81.6
	AdamW	R@1: 30.8	**R@1: 49.2**
		R@5: 44.7	R@5: 68.6
		R@10: 51.8	R@10: 75.9
		R@20: 57.3	R@20: 81.6
NetVLAD	Adam	R@1: 22.8	R@1: 36.2
		R@5: 39.0	R@5: 55.2
		R@10: 44.7	**R@10: 64.4**
		R@20: 51.0	R@20: 72.7
	AdamW	**R@1: 23.3**	**R@1: 37.1**
		R@5: 35.6	**R@5: 57.5**
		R@10: 43.0	R@10: 63.2
		R@20: 50.6	**R@20: 70.2**

Table 7. Optimizers performance

domain adaptation. However, performance on the *Tokyo_XS* dataset did not show a comparable improvement. This is likely due to the ineffectiveness of domain shifting when using *St_Lucia* as the target, possibly attributed to the dataset's limited size.

4.3. Optimizers

In the final set of experiments, we evaluated the performance of the optimizers Adam and AdamW using both the aggregators NetVLAD and MixVPR. For these tests, the learning rate was set to e^{-5} and the weight decay to 0.001. As indicated in Table 7, the results did not show a significant improvement in performance, leading to inconclusive findings.

5. Conclusion

In this research, we extend the capabilities of the CosPlace framework by exploring advancements in data augmentation, feature aggregation, domain adaptation, and optimization techniques. Our findings indicate that enhancing image contrast in data augmentation improves the sensitivity of the images from the *SanFrancisco_XS* dataset, thereby boosting recall performance.

Empirical evaluations on both the *SanFrancisco_XS* and *Tokyo_XS* datasets demonstrate that MixVPR serves as a superior feature aggregation technique, as corroborated by the results in Table 4. Further, the employment of Adversarial Domain Adaptation in conjunction with NetVLAD yields significant performance improvements, especially on the *SanFrancisco_XS* dataset, as shown in Table 5.

Unfortunately, our investigations into various optimizers did not yield the impactful contributions to this paper that we had hoped for.

Despite notable progress, certain aspects of the study remain underexplored due to limitations like computational resources and time constraints. For instance, future research could delve into the impact of combining schedulers with optimizers like Adam and AdamW. Additionally, re-evaluating the ASGD optimizer with a higher number of iterations could be insightful. Another intriguing avenue could be assessing the potential advantages of incorporating MixVPR as an aggregator within a domain adaptation framework. Our study also has the limitation of relying on just one large dataset for training and a single backbone architecture. These are all points that could be explored in future research.

References

[1] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. 2012. 1

[2] Brahim Chaib-draa Amar Ali-bey and Philippe Giguere. Mixvpr: Feature mixing for visual place recognition. 2023. 2, 4

[3] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016. 1, 2

[4] R. Arandjelovic and A. Zisserman. All about vlad. CVPR, 2013. 1

[5] Gabriele Berton, Carlo Masone, and Barbara Caputo. Rethinking visual geo-localization for large-scale applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4878–4888, June 2022. 1, 2, 4

[6] Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations, San Diego, 2015*, December 2014. 2

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1180–1189, 2015. 2

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[9] Frank Hutter Ilya Loshchilov. Fixing weight decay regularization in adam. February 2018. 2

[10] Frank Hutter Ilya Loshchilov. Decoupled weight decay regularization. *ICLR*, 2019. 2

[11] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Shorten and Khoshgoftaar J Big Data*, 2019. 1

[12] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 2

[13] Wei Xu. Towards optimal one pass large scale learning with averaged stochastic gradient descent. July 2011. 2

[14] Lili Mou Ge Li Yunchuan Chen Yangyang Lu Zhi Jin Yan Xu, Ran Jia. Improved relation classification by deep recurrent neural network with data augmentation. *Key Laboratory of High Confidence Software Technologies (Peking University)*, 2016. 1