On Feature Normalization and Data Augmentation (Supplementary Material)

Boyi Li*[12] Felix Wu*[3] Ser-Nam Lim[4] Serge Belongie[12] Kilian Q. Weinberger[13]
[1]Cornell University [2]Cornell Tech [3]ASAPP [4]Facebook AI
{bl728,sjb344,kilian}@cornell.edu fwu@asapp.com sernamlim@fb.com

A. MoEx PyTorch Implementation

Algorithm 1 shows an example code of MoEx in PyTorch [5].

```
# x: a batch of features of shape (batch_size,
#    channels, height, width),
# y: onehot labels of shape (batch_size, n_classes)
# norm_type: type of the normalization to use

def moex(x, y, norm_type):
    x, mean, std = normalization(x, norm_type)
    ex_index = torch.randperm(x.shape[0])
    x = x * std[ex_index] + mean[ex_index]
    y_b = y[ex_index]
    return x, y, y_b

# output: model output
# y: original labels
# y_b: labels of moments
# loss_func: loss function used originally
# lam: interpolation weight $\lambda$
def interpolate_loss(output, y, y_b, loss_func, lam):
    return lam * loss_func(output, y) + \
        (1. - lam) * loss_func(output, y_b)

def normalization(x, norm_type, epsilon=1e-5):
    # decide how to compute the moments
    if norm_type == 'pono':
        norm_dims = [1]
    elif norm_type == 'instance_norm':
        norm_dims = [2, 3]
    else: # layer norm
        norm_dims = [1, 2, 3]
    # compute the moments
    mean = x.mean(dim=norm_dims, keepdim=True)
    var = x.var(dim=norm_dims, keepdim=True)
    std = (var + epsilon).sqrt()
    # normalize the features, i.e., remove the moments
    x = (x - mean) / std
    return x, mean, std
```

Algorithm 1. Example code of MoEx in PyTorch.

B. MoEx for NLP

B.1. Machine Translation on IWSLT 2014

To show the potential of MoEx on natural language processing (NLP) tasks, we apply MoEx to the state-of-the-art DynamicConv [6] model on 4 tasks in a benchmarking dataset IWSLT 2014 [1]: German to English, English to German, Italian to English, and English to Italian machine translation. IWSLT 2014 is based on the transcripts of TED talks and their translation, it contains 167K English and German sentence pairs and 175K English and Italian sentence pairs. We use fairseq library [3] and follow the common setup [2] using 1/23 of the full training set as the validation set for hyper-parameter selection and early stopping. All models are trained with a batch size of 12000 tokens per GPU on 4 GPUs for 20K updates to ensure convergence; however, the models usually don't improve after 10K updates. We use the validation set to select the best model. We tune the hyper-parameters of MoEx on the validation set of the German to English task including $p \in \{0.25, 0.5, 0.75, 1.0\}$ and $\lambda \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ and use MoEx with InstanceNorm with $p = 0.5$ and $\lambda = 0.8$ after the first encoder layer. We apply the same set of hyper-parameters to the other three language pairs. When computing the moments, the edge paddings are ignored. We use two metrics to evaluate the models: BLEU [4] which is a exact word-matching metric and scaled BERTScore F1 [7].

Table 1 summarizes the average scores (higher better) with standard error rates over three runs. It shows that MoEx consistently improves the baseline model on all four tasks by about 0.2 BLEU and 0.2% BERT-F1. Although these improvements are not exorbitant, they are highly consistent and, as far as we know, MoEx is the first label-perturbing data augmentation method that improves machine translation models.

C. More Examples of MoEx

Figure 1 shows more examples of MoEx. We select top five features out of 64 channels to show here.

References

[1] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014. In Proceedings of the International Workshop on Spoken Language Translation, 2014. 1

[2] Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. Classical structured prediction losses for sequence to sequence learning. In NAACL-HLT, 2018. 1

[3] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan,

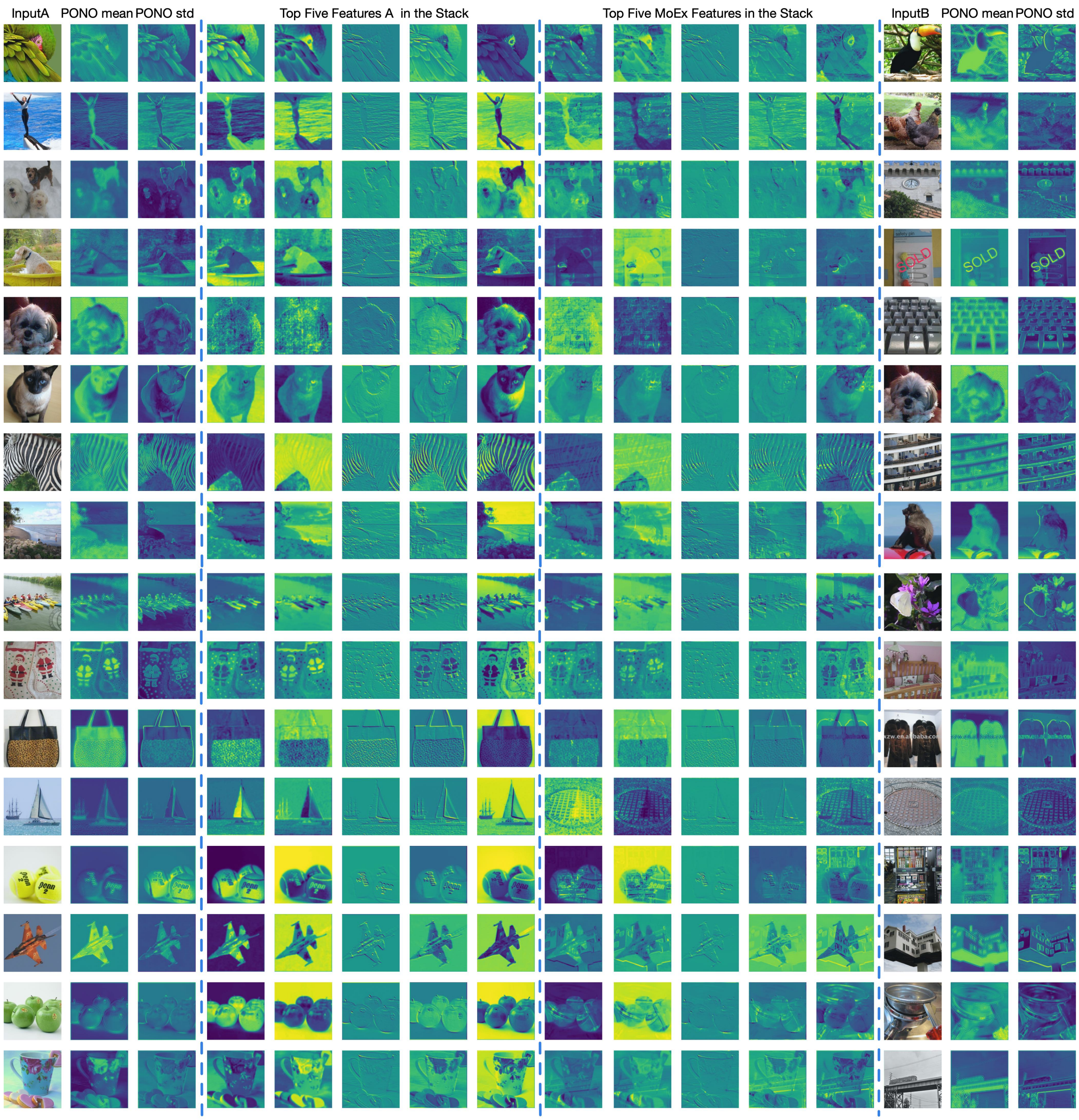

Figure 1. MoEx with PONO normalization. The features of image A are normalized and then infused with moments $\boldsymbol{\mu}_B$ (PONO mean), $\boldsymbol{\sigma}_B$ (PONO std) from the image B.

Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In NAACL-HLT), 2019. 1

[4] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In ACL, 2002. 1

[5] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 1

[6] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic

Task	Method	BLEU ↑	BERT-F1 (%) ↑
De-En	Transformer	34.4†	-
	DynamicConv	35.2†	-
	DynamicConv	35.46±0.06	67.28±0.02
	+ MoEx	**35.64±0.11**	**67.44±0.09**
En-De	DynamicConv	28.96±0.05	63.75±0.04
	+ MoEx	**29.18±0.10**	**63.86±0.02**
It-En	DynamicConv	33.27±0.04	65.51±0.02
	+ MoEx	**33.36±0.11**	**65.65±0.07**
En-It	DynamicConv	30.47±0.06	64.05±0.01
	+ MoEx	**30.64±0.06**	**64.21±0.11**

Table 1. Machine translation with DynamicConv [6] on IWSLT-14 German to English, English to German, Italian to English, and English to Italian tasks. The mean and standard error are based on 3 random runs. †: numbers from [6]. Note: for all these scores, the higher the better.

convolutions. In ICLR, 2019. 1, 3

[7] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In ICLR, 2020. 1