7 Supplement

7.1 TypeNet Configuration

Figure 5 shows the data flow for our model as configured for the All-Pairs problem. The Tables 1, 2, and 3 present the detailed network configuration (also found in the sample code distributed with the dataset generator):

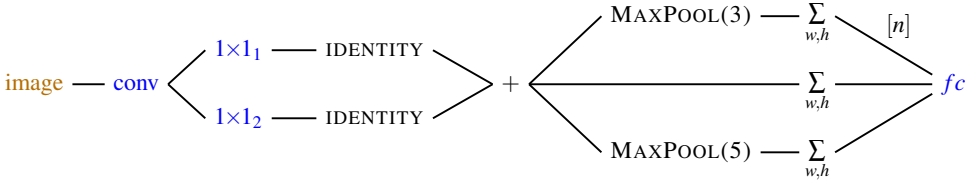

Figure 5: Model data-flow used for All-Pairs.

parameter	value
N_c	4
N_f	4
N_t	2
N_s	3
n	64
AC	ELU
A	IDENTITY
SPATIAL	{IDENTITY, MAXPOOL3X3, MAXPOOL5X5}
AFC	ELU

Table 1: Model level parameters for TypeNet as used to solve All-Pairs.

For the activation, A_i, the most useful activations were found to be IDENTITY, SELU, and SOFTMAX. SOFTMAX is in the feature, rather than spatial dimension. Via architecture search, IDENTITY is the most generally useful activation, though SOFTMAX tended to reduce training times (probably because it forms a strong approximately-sparse bottleneck).

The convolution block used ELU activation, no bias, batch norm (post-activation), and padding to align the convolution filters with the image edges. It's layer-wise characteristics are detailed below. For larger images, a stride of 2 was used in CONV_3.

parameter	features, size, stride
CONV_1	128, 3, 1
CONV_2	128, 5, 2
CONV_3	128, 5, 1
CONV_4	128, 3, 1

Table 2: Convolution block parameters for TypeNet as used to solve All-Pairs.

The fully-connected layers have input of size $m = N_s \times n$ and their configuration is detailed below:

parameter	value
fc_1	m-ELU-bnorm
fc_2	$\lfloor \frac{m}{2} \rfloor$-ELU-bnorm
fc_3	$\lfloor \frac{m}{4} \rfloor$-ELU-bnorm
fc_4	2-IDENTITY

Table 3: Fully-connected parameters for TypeNet as used to solve All-Pairs.

7.2 TypeNet Architecture Search

The main hyper-parameters and architecture-variations explored are the feature activation, number of branches (k), and number of features (n). First, we explored the choice of activation with $n = 64$ and $k = 2$. All activation combinations drawn from the following options were explored and the top results are presented in Figure 6 : ELU (E), IDENTITY (I), RELU (R), SELU (Se), SIGMOID (S), SOFTMAX (Sm), SOFTPLUS (Sp), and TANH (T). In each figure, architectures are labeled with n when $n \neq 64$, and the above abbreviations of the k activations are used. If a "-w" is appended, the architecture had a wider convolution receptive field (the stride of the third **conv** layer was 2).

$$[n\text{-}]\mathsf{Activation}_1[...\mathsf{Activation}_k][\text{-w}]. \tag{1}$$

All of the runs represented in Figure 6 had higher accuracy than any of the baselines. The main conclusion from these trials is that SOFTMAX and SELU are the most useful activations. We most frequently used SOFTMAX as the activation in exploring the other hyper-parameters because of its low training variance.

We studied how the number of branches, k, affects training; those results are shown below with the number of training samples needed to fully solve the 4-4 All-Pairs problem. All trials reached 100% accuracy, save for one three-branch trial which got stuck at a test accuracy of 99.948% after 30M training examples. Based on the number of samples needed to reach maximum test accuracy, we conclude that $k = 2$ is best for this problem.

branches (k)	accuracy	training samples
1 [×9]	1.0 ± 0.0	$57.1M \pm 3.8M$
2 [×10]	1.0 ± 0.0	$47.7M \pm 4.7M$
3 [×20]	1.0 ± 10^{-4}	$49.4M \pm 8.9M$

The SOFTMAX activated network with two branches was found to train faster for more features as summarized in the following table:

features (n)	accuracy	training samples
48 [×9]	1.0 ± 0.0	$57.0M \pm 8.9M$
64 [×10]	1.0 ± 0.0	$47.7M \pm 4.7M$
96 [×20]	1.0 ± 0.0	$40.5M \pm 7.7M$.

All options consistently achieved 100% test accuracy, so this trade-off for the 4-4 problem can be made to optimize training time or inference time.

7.3 More Details on the Harder All-Pairs Problems

The TypeNet approach cannot easily be made to solve every All-Pairs problem; Figure 7 shows results for the 5-5, 6-6, and 7-7 All-Pairs problem. The IDENTITY activation was the

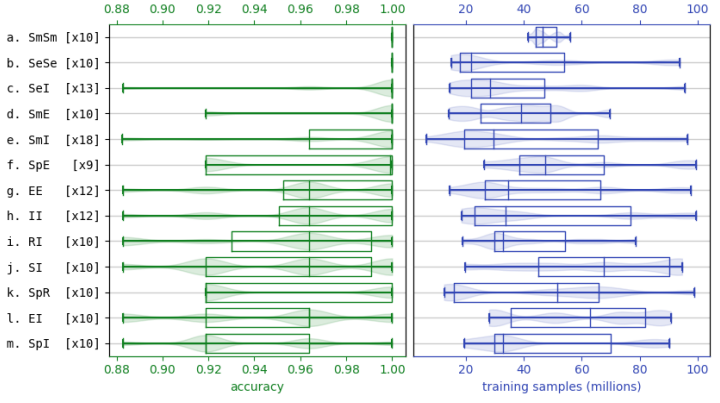

Figure 6: 4-4 All-Pairs for different activation functions, A_i.

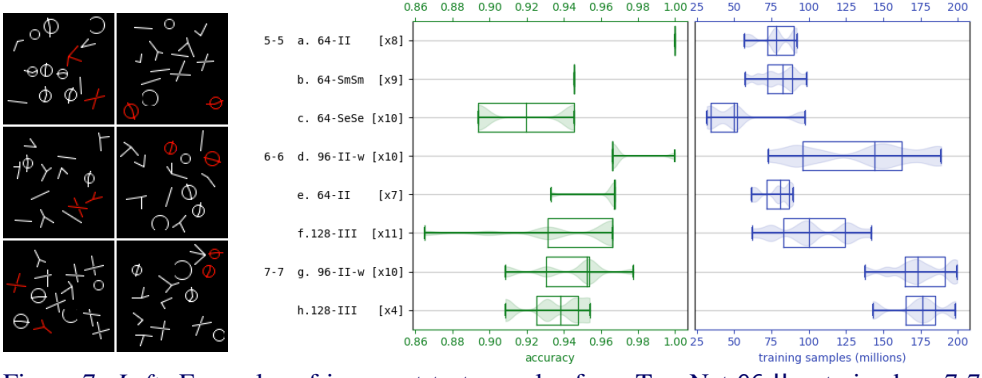

Figure 7: *Left*: Examples of incorrect test samples from TypeNet 96-II-w trained on 7-7 All-Pairs for 200M samples. White symbols can be paired, leaving the red symbols unpaired. *Right*: Test results of applying TypeNet to more difficult All-Pairs problems. Wider **conv** receptive fields are notated with "-w", see text for details.

only activation to reach 100% accuracy on the 5-5 and 6-6 problem, in 100% (Fig7-a) and 20% (Fig7-d) of trials respectively. The SELU and SOFTMAX activation were not successful on any of these problems in any trail within the 100M training sample limit.

For these problems, the image size was increased from 76×76 to 96×96 to make room for all the symbols. This image size increase required decreasing the batch size from 600 to 400; all other training settings remained unchanged. The large image size led us to expand the receptive field of the **conv** as notated with "-w" and detailed in Section 7.2. The most enlightening observations from these experiments are as follows:

- The SELU activation (Fig7-c) had lower accuracy than expected from its effectiveness on the 4-4 problem.
- On these harder problems, the SOFTMAX activation continued to show lower variance across trials in both accuracy and training samples.
- The SmSm model (Fig7-b) consistently got stuck at 94.6% accuracy on the 5-5 problem, perhaps because the SOFTMAX activations are prone to local minima.
- The number of branches was increased to 3 and number of features to 128, indepen-

dently and together, for the best case activations from smaller models. The 128-III (Fig7-f) model had the best test accuracy, but did worst than the simpler II model (Fig7-e) even when trained to 200M training examples.

- The 7-7 All-Pairs problem (Fig7-g,h) is clearly harder. The wider 96-II-w (Fig7-g) model was the best.
- As shown in Figure 7-Left, the test samples missed by one of the 96-II-w models on the 7-7 problem are semantically similar: the model incorrectly labels some samples as *true* that have either an unpaired **cross** and **3-star**, or an unpaired **theta** and **phi**. For this model and trial, all of its errors fall into these two classes, though it correctly classifies some of those examples (achieving a 95% accuracy when those two classes account for 9.5% of the test set). Different trials show different types of semantic errors.
- Many variations, including mixtures of activations, more features, more branches, even wider **conv** receptive fields, and combinations of these choices, were tried to solve the 7-7 problem without success. In the highest test accuracy observed (98%), the misclassified images are still easy for a human to classify.

7.4 Comparison to a Simple CNN

Are Lines 5 and 6 of Algorithm 1 generally useful, and do they improve the algorithm? The table below compares (3 trials for each) the test accuracy and model size of TypeNet with a with a simple convolutional net (ConvNet) created by altering TypeNet as follows:

- Replace Lines 5 and 6 of Algorithm 1 with FLATTEN (passing the convolution output directly to the fully-connected layers).
- As with larger All-Pairs images, use a stride of 2 in CONV_3.

	ConvNet		TypeNet	
dataset	accuracy	# parameters	accuracy	# parameters
MNIST	0.9953 ± 0.0002	2M	$\mathbf{0.9971 \pm 0.0006}$	**1M**
Fashion-MNIST	$\mathbf{0.9409 \pm 0.0005}$	2M	0.9346 ± 0.0011	**1M**
CIFAR10	0.7773 ± 0.0013	2.5M	$\mathbf{0.8820 \pm 0.0080}$	**1M**
4-4 All-Pairs	0.8080 ± 0.0925	9.9M	$\mathbf{1.0000 \pm 0.0000}$	**1M**

From this comparison, TypeNet is seen to have fewer parameters and shows significant improvements in accuracy for the hardest two datasets (CIFAR10 and 4-4 All-Pairs). The number of parameters in TypeNet is not dependent on the input size because of the spatial summation in Line 6 of the algorithm. We anticipate the spatial, learned histogram of TypeNet to be a useful tool in the construction of other DNN architectures.

7.5 Observations

Most existing gradient based WSL models (including TypeNet) require processing millions of training samples (see Figure 3, right). Humans on the other hand, learn from few labels; reproducing this efficiency is a valuable goal and likely will mark notable progress in the field of machine learning. Recent work [22] examined the SNR of the gradients in simple binary classification problems and concluded that the reason for the slow convergence of WSL objectives was low signal strength in the loss gradient, $\mathbb{E}[\nabla\mathcal{L}]$, coupled with a high

variance, $Var[\nabla\mathcal{L}]$. In contrast, a method using k task-specific divisions of the same problem had $\frac{1}{k}$ less variance. Future work will attempt to reduce the variance of TypeNet gradients using variance reduction techniques such as control variates [20] and other similar methods.

The current TypeNet models uses simple similarity features and requires the practitioner to tune the receptive field size of the *types* (learned histogram features) to the problem. Future work can address these limitations using some of the following methods:

- Use more complicated base feature extractors such as using ResNet [7] or InceptionV3 [28].
- Use a multi-scale receptive field configuration and learn a mixture of these scales.
- Use learned saliency masks to scale the contribution from different regions.