Learning to Compose Domain-Specific Transformations for Data Augmentation
Alexander J. Ratner*, Henry R. Ehrenberg*, Zeshan Hussein, Jared Dunnmon, Christopher Ré

Data augmentation: a critical (hand-tuned) tool
Data augmentation is the technique of enlarging training sets with class-preserving transformations—a form of weak supervision.

Encoding knowledge in multiple domains
Images
- Rotations
- Crops
- Hue shifts
- Brightness shifts

NLP
- Synonym swaps
- Language model swaps
- Positional swaps
- Standard image TFS
- Parametrized value shifts
- Move segmented mass
- Texture blending

Medical

Many more transformations than just e.g. rotations and crops—but complex transformations are more difficult to tune and compose! New domains are tough to spin up, too.

Augmentation as sequence modeling
We represent data augmentation pipelines as sequences of incremental transformation operations. This is a simple (discretized) way to parameterize and compose transformations.

Weakening the invariance assumption
We make the key assumption that TFS will either preserve class, or map to a “null” class, but will not map between classes. This allows us to train with unlabeled data!

Robustness to misspecification
A major advantage of our approach is that we learn to avoid misspecified transformation functions by the user.

Performance on multi-domain datasets
We improve performance on MNIST, CIFAR-10, text relation extraction (ACE), and mammography classification (DDSM).

Users write transformation functions (TFS)

We learn a sequence model adversarially to generate sequences of TFSs

Use for standard data augmentation with any end discriminative model

TF sequences

Trained Generator

Labeled Data

$D_f$

Trained End Model

Unlabeled Data

$D_θ$

$G$

$G$

$G$

$G$

Independent model (MF)
Learns TF sampling frequencies, samples independently of previously applied TFSs.

State-based model (LSTM)
Learns to compose TFSs in specific orders.

We make the key assumption that TFSs will either preserve class, or map to a “null” class, but will not map between classes. This allows us to train with unlabeled data!

$y(h_{T_2} \circ \cdots \circ h_1(x)) \in \{y(x), y_θ\}$

$\mathcal{G}_θ$:

$\mathcal{G}_θ$:

$\mathcal{G}_θ$:

$\mathcal{G}_θ$:

Independent model (MF)

State-based model (LSTM)

We improve performance on MNIST, CIFAR-10, text relation extraction (ACE), and mammography classification (DDSM).

End Model Classification Results: Random Subsample

<table>
<thead>
<tr>
<th>Task</th>
<th>% None</th>
<th>Basic</th>
<th>Heur.</th>
<th>MF</th>
<th>LSTM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>1</td>
<td>90.2</td>
<td>95.3</td>
<td>95.9</td>
<td>96.5</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>1</td>
<td>97.3</td>
<td>98.7</td>
<td>99.0</td>
<td>99.2</td>
</tr>
<tr>
<td>ACE (P1)</td>
<td>100</td>
<td>66.0</td>
<td>73.1</td>
<td>77.5</td>
<td>79.8</td>
</tr>
<tr>
<td>DDSM</td>
<td>1</td>
<td>57.6</td>
<td>58.8</td>
<td>59.3</td>
<td>58.2</td>
</tr>
<tr>
<td>DDSM + DS</td>
<td>10</td>
<td>57.6</td>
<td>58.8</td>
<td>59.3</td>
<td>58.2</td>
</tr>
</tbody>
</table>

Adversarial objective optimized by policy gradient descent:

$$J_θ = E_{x \sim G_θ} E_{x' \sim \tilde{U}} [\log(1 - D_θ(h_{T_2} \circ \cdots \circ h_1(x)))] + E_{x' \sim \tilde{U}} [\log(D_θ(x'))]$$

Users write transformation functions (TFSs)

We learn a sequence model adversarially to generate sequences of TFSs

Use for standard data augmentation with any end discriminative model

Robustness to misspecification

A major advantage of our approach is that we learn to avoid misspecified transformation functions by the user.

Misspecified TFSs (red) are sampled with very low frequency as training progresses—the user does not need to be perfect!