CS294-158 Deep Unsupervised Learning

Lecture 9:

(a) Semi-Supervised Learning

(b) Unsupervised Distribution Alignment

Pieter Abbeel, Xi (Peter) Chen, Jonathan Ho, Aravind Srinivas, Alex Li, Wilson Yan

UC Berkeley

Covid and Mid-Semester Update

- We hope everyone and their families are able to keep healthy during these unusual times. Please prioritize your health and well-being! Accordingly, please don't hesitate to let us know if this class would interfere with that, and we'll be happy to figure something out on a case by case basis!

- Since we are just past the middle of the semester, and since there is some re-planning per the current situation, here is a quick overview of what's still ahead:

- 4/1: **L9** Semi-Supervised Learning; Unsupervised Distribution Alignment [this lecture! :)]
- 4/8: **L10** Compression
- 4/13: **Final Project 3-page milestone**: instructions
- 4/15: **L11** Language Models [Guest Instructor: Alec Radford (OpenAI)]
- 4/22: **Midterm**
- 4/29: **L12** Representation Learning in Reinforcement Learning
- 5/6: RRR Week (no lecture)
- 5/13: **Final Project Presentations + Final Project Reports** due

Outline

- Intro
- ***Semi-Supervised Learning***
- Unsupervised Distribution Alignment

What is Semi-Supervised Learning?

Supervised Learning

What is Semi-Supervised Learning?

Supervised Learning

$$(x, y) \sim p(x, y)$$

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)} \left[\log p(y|x) \right]$$

What is Semi-Supervised Learning?

Supervised Learning

$$(x, y) \sim p(x, y)$$

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)} \left[\log p(y|x) \right]$$

Semi-Supervised Learning

What is Semi-Supervised Learning?

Supervised Learning

$$(x, y) \sim p(x, y)$$

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)} \left[\log p(y|x) \right]$$

Semi-Supervised Learning

$$D_U : x \sim p(x), D_S : (x, y) \sim p(x, y)$$

What is Semi-Supervised Learning?

Supervised Learning

$$(x, y) \sim p(x, y)$$

$$\max \quad \mathbb{E}_{x,y \sim p(x,y)} \left[\log p(y|x) \right]$$

Semi-Supervised Learning

$$D_U : x \sim p(x), D_S : (x, y) \sim p(x, y)$$

Semi-Supervised Learning

- Classification: Fully Supervised
  - Training data: (image, label), predict label for new images.

- What if we have a few labeled samples and many unlabeled samples? Labeling is generally time-consuming and expensive in certain domains.

- Semi-Supervised Learning
  - Training data: Labeled data (image, label) and Unlabeled data (image)
  - **Goal: Use the unlabeled data to make supervised learning better**

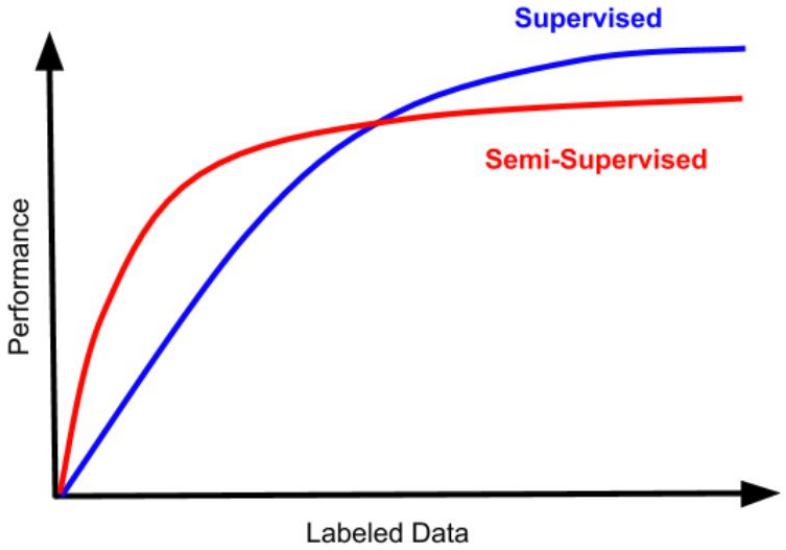
Why Semi-Supervised Learning?

Labeled Data

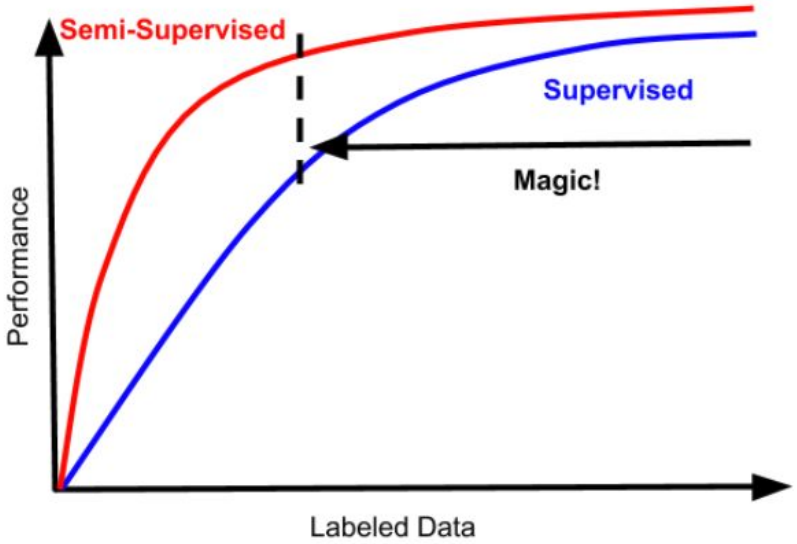
Unlabeled Data

Slide: Thang Luong

Why Semi-Supervised Learning?



Belief of many ML practitioners



Dream of many SSL researchers

Slide: Thang Luong

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - UDA, MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - Unsupervised Data Augmentation (UDA), MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Agenda

- Core concepts
  - Confidence vs Entropy
    - Entropy minimization
    - Pseudo Labeling
    - Virtual Adversarial Training
  - Label Consistency
    - Make sure augmentations of the sample have the same class
    - Pi-Model, Temporal Ensembling, Mean Teacher
  - Regularization
    - Weight decay
    - Dropout
    - Data-Augmentation (MixUp, CutOut)
    - Unsupervised Data Augmentation (UDA), MixMatch
  - Co-Training / Self-Training / Pseudo Labeling (Noisy Student)

Entropy Minimization

Semi-supervised Learning by Entropy Minimization

Yves Grandvalet *
Heudiasyc, CNRS/UTC
60205 Compiègne cedex, France
grandval@utc.fr

Yoshua Bengio
Dept. IRO, Université de Montréal
Montreal, Qc, H3C 3J7, Canada
bengioy@iro.umontreal.ca

Abstract

We consider the semi-supervised learning problem, where a decision rule is to be learned from labeled and unlabeled data. In this framework, we motivate minimum entropy regularization, which enables to incorporate unlabeled data in the standard supervised learning. Our approach includes other approaches to the semi-supervised problem as particular or limiting cases. A series of experiments illustrates that the proposed solution benefits from unlabeled data. The method challenges mixture models when the data are sampled from the distribution class spanned by the generative model. The performances are definitely in favor of minimum entropy regularization when generative models are misspecified, and the weighting of unlabeled data provides robustness to the violation of the "cluster assumption". Finally, we also illustrate that the method can also be far superior to manifold learning in high dimension spaces.

Pseudo Labeling

Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks

Dong-Hyun Lee SAYIT78@GMAIL.COM
Nangman Computing, 117D Garden five Tools, Munjeong-dong Songpa-gu, Seoul, Korea

Abstract

We propose the simple and efficient method of semi-supervised learning for deep neural networks. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Labels*, just picking up the class which has the maximum predicted probability, are used as if they were true labels. This is in effect equivalent to *Entropy Regularization*. It favors a low-density separation between classes, a commonly assumed prior for semi-supervised learning. With Denoising Auto-Encoder and Dropout, this simple method outperforms conventional methods for semi-supervised learning with very small labeled data on the MNIST handwritten digit dataset.

and unsupervised tasks using same neural network simultaneously. In (Ranzato et al., 2008), the weights of each layer are trained by minimizing the combined loss function of an autoencoder and a classifier. In (Larochelle et al., 2008), *Discriminative Restricted Boltzmann Machines* model the joint distribution of an input vector and the target class. In (Weston et al., 2008), the weights of all layers are trained by minimizing the combined loss function of a global supervised task and a *Semi-Supervised Embedding* as a regularizer.

In this article we propose the simpler way of training neural network in a semi-supervised fashion. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Labels*, just picking up the class which has the maximum predicted probability every weights update, are used as if they were true la-

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Confidence vs Entropy

- Consider image x, classes {y1, y2, y3}.
- Classifier A: Probabilities for y: [0.1, 0.8, 0.1]
- Classifier B: Probabilities for y: [0.1, 0.6, 0.3]
- Classifier A is **more confident and has lower entropy.**
- Entropy minimization akin to training on confident predictions as pseudo labels.

Label Consistency with Data Augmentation

Label Consistency with Data Augmentation

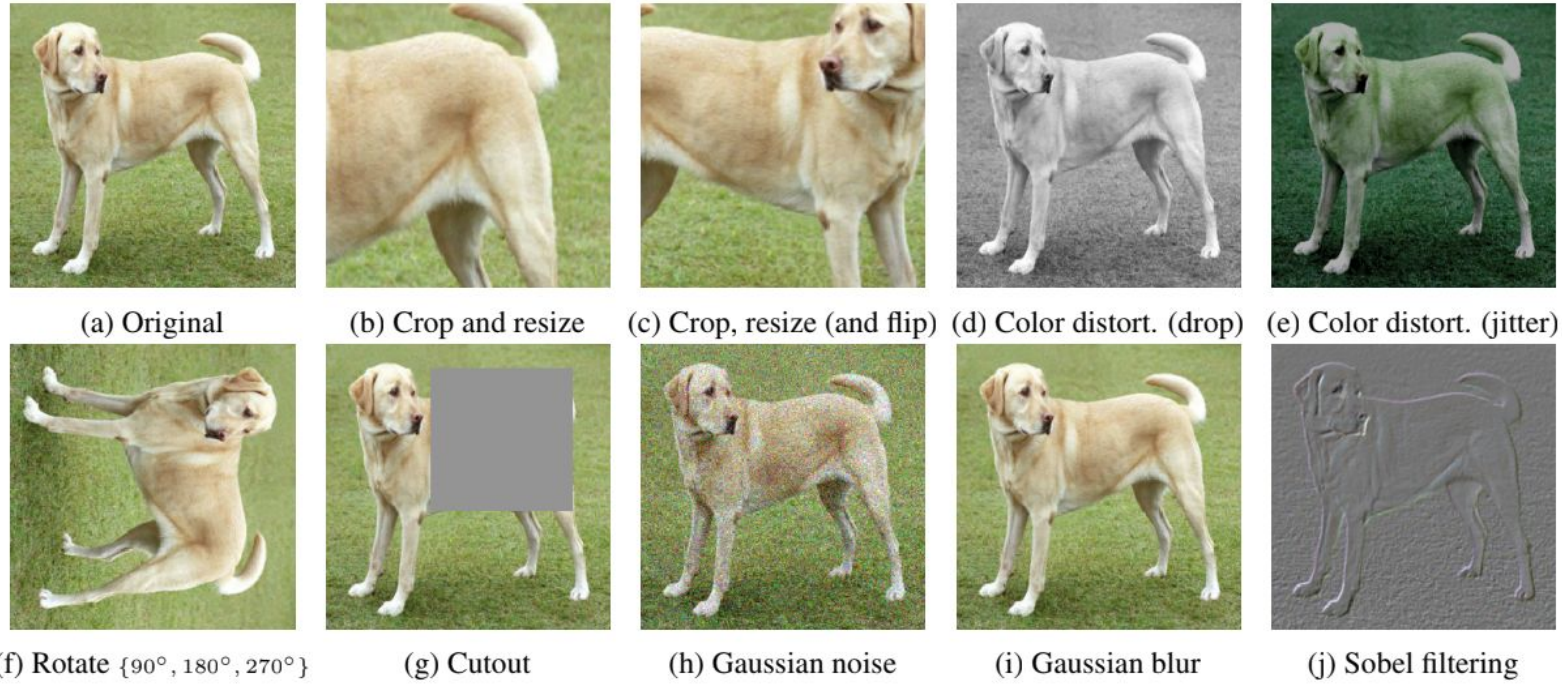
Could be Unlabeled or Labeled

Label Consistency with Data Augmentation

Label Consistency with Data Augmentation

Make sure that the logits are similar

More Data Augmentation -> Regularization



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Realistic Evaluation of Semi-Supervised Learning

Realistic Evaluation of Deep Semi-Supervised Learning Algorithms

Avital Oliver,* Augustus Odena,* Colin Raffel,* Ekin D. Cubuk & Ian J. Goodfellow
Google Brain
{avitalo,augustusodena,craffel,cubuk,goodfellow}@google.com

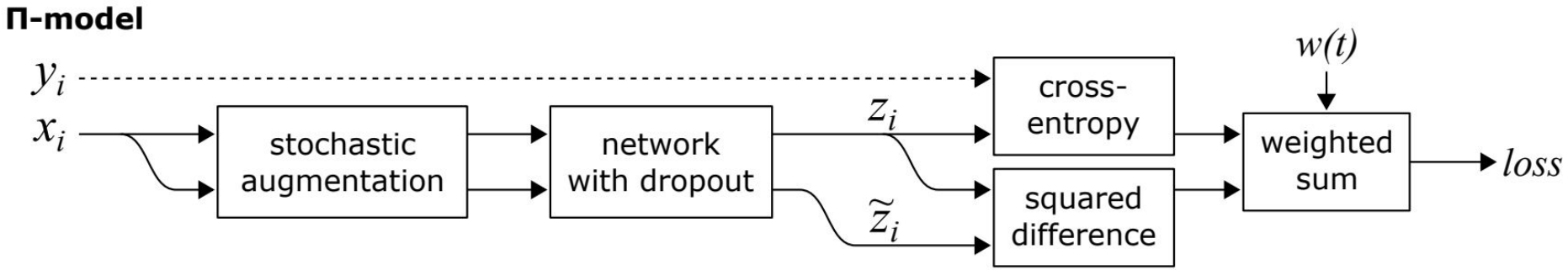
Abstract

Semi-supervised learning (SSL) provides a powerful framework for leveraging unlabeled data when labels are limited or expensive to obtain. SSL algorithms based on deep neural networks have recently proven successful on standard benchmark tasks. However, we argue that these benchmarks fail to address many issues that SSL algorithms would face in real-world applications. After creating a unified reimplementation of various widely-used SSL techniques, we test them in a suite of experiments designed to address these issues. We find that the performance of simple baselines which do not use unlabeled data is often underreported, SSL methods differ in sensitivity to the amount of labeled and unlabeled data, and performance can degrade substantially when the unlabeled dataset contains out-of-distribution examples. To help guide SSL research towards real-world applicability, we make our unified reimplemention and evaluation platform publicly available.[2]

Outline

- Realistic Evaluation of Semi-Supervised Learning
  - pi-model
  - Temporal Ensembling
  - Mean Teacher
  - Virtual Adversarial Training

pi-Model



Temporal Ensembling for Semi-Supervised Learning

pi-Model

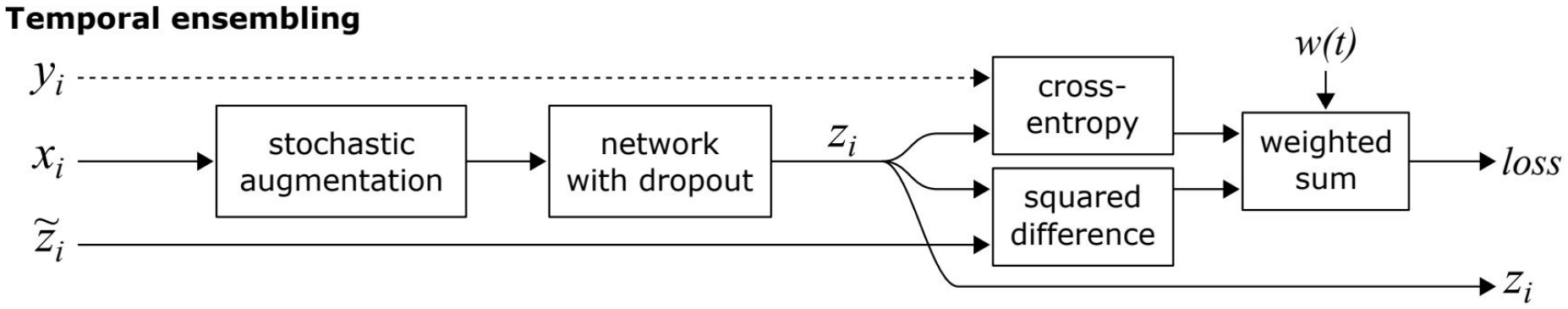
Algorithm 1 Π-model pseudocode.

Require: x_i = training stimuli
Require: L = set of training input indices with known labels
Require: y_i = labels for labeled inputs $i \in L$
Require: $w(t)$ = unsupervised weight ramp-up function
Require: $f_\theta(x)$ = stochastic neural network with trainable parameters θ
Require: $g(x)$ = stochastic input augmentation function

for t in $[1, num\_epochs]$ **do**
  for each minibatch B **do**
    $z_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ evaluate network outputs for augmented inputs
    $\tilde{z}_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ again, with different dropout and augmentation
    $loss \leftarrow -\frac{1}{|B|} \sum_{i \in (B \cap L)} \log z_i[y_i]$ ▷ supervised loss component
      $+ w(t) \frac{1}{C|B|} \sum_{i \in B} ||z_i - \tilde{z}_i||^2$ ▷ unsupervised loss component
    update θ using, e.g., ADAM ▷ update network parameters
  end for
end for
return θ

Temporal Ensembling for Semi-Supervised Learning

Temporal Ensembling



Temporal Ensembling for Semi-Supervised Learning

Temporal Ensembling

Algorithm 2 Temporal ensembling pseudocode. Note that the updates of Z and $\tilde{z}$ could equally well be done inside the minibatch loop; in this pseudocode they occur between epochs for clarity.

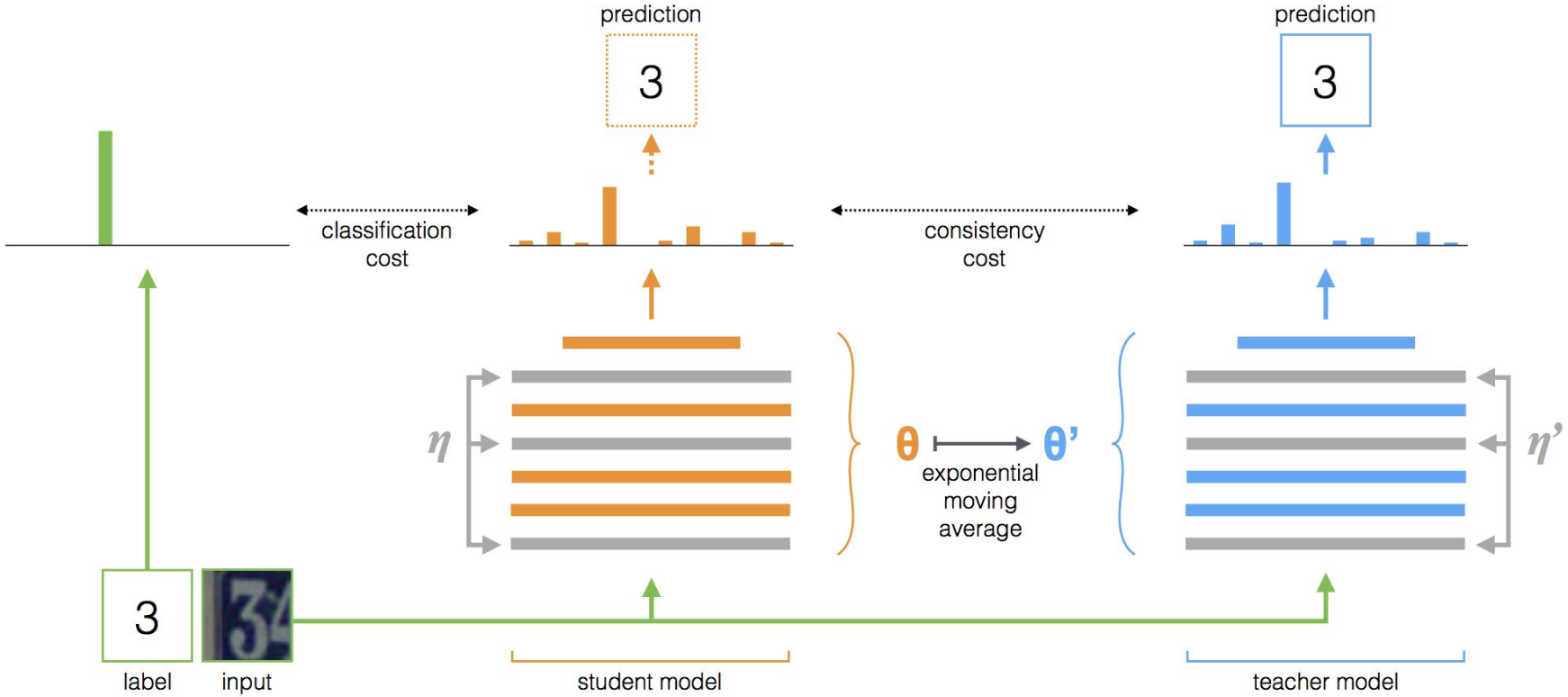
Require: x_i = training stimuli
Require: L = set of training input indices with known labels
Require: y_i = labels for labeled inputs $i \in L$
Require: α = ensembling momentum, $0 \leq \alpha < 1$
Require: $w(t)$ = unsupervised weight ramp-up function
Require: $f_\theta(x)$ = stochastic neural network with trainable parameters θ
Require: $g(x)$ = stochastic input augmentation function

$Z \leftarrow \mathbf{0}_{[N \times C]}$ ▷ initialize ensemble predictions
$\tilde{z} \leftarrow \mathbf{0}_{[N \times C]}$ ▷ initialize target vectors
for t in $[1, num\_epochs]$ **do**
  for each minibatch B **do**
    $z_{i \in B} \leftarrow f_\theta(g(x_{i \in B}, t))$ ▷ evaluate network outputs for augmented inputs
    $loss \leftarrow -\frac{1}{|B|} \sum_{i \in (B \cap L)} \log z_i[y_i]$ ▷ supervised loss component
      $+ w(t) \frac{1}{C|B|} \sum_{i \in B} ||z_i - \tilde{z}_i||^2$ ▷ unsupervised loss component
    update θ using, e.g., ADAM ▷ update network parameters
  end for
  $Z \leftarrow \alpha Z + (1 - \alpha) z$ ▷ accumulate ensemble predictions
  $\tilde{z} \leftarrow Z / (1 - \alpha^t)$ ▷ construct target vectors by bias correction
end for
return θ

Temporal Ensembling for Semi-Supervised Learning

Mean Teacher

Mean Teachers are better role models



Virtual Adversarial Training

$$\boldsymbol{r}_{\text{adv}} = -\epsilon \boldsymbol{g} / \|\boldsymbol{g}\|_2 \text{ where } \boldsymbol{g} = \nabla_{\boldsymbol{x}} \log p(y \mid \boldsymbol{x}; \hat{\boldsymbol{\theta}})$$

$$\boldsymbol{r}_{\text{v-adv}} = \underset{\boldsymbol{r}, \|\boldsymbol{r}\| \leq \epsilon}{\arg\max} \mathrm{KL}[p(\cdot \mid \boldsymbol{x}; \hat{\boldsymbol{\theta}}) || p(\cdot \mid \boldsymbol{x} + \boldsymbol{r}; \hat{\boldsymbol{\theta}})]$$

Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning

Virtual Adversarial Training

Algorithm 1 Mini-batch SGD for $\nabla_\theta \mathcal{R}_{\text{vadv}}(\theta)|_{\theta=\hat{\theta}}$, with a one-time power iteration method.

1) Choose M samples of $x^{(i)}(i = 1, \ldots, M)$ from dataset $\mathcal{D}$ at random.
2) Generate a random unit vector $d^{(i)} \in R^I$ using an iid Gaussian distribution.
3) Calculate r_{vadv} via taking the gradient of D with respect to r on $r = \xi d^{(i)}$ on each input data point $x^{(i)}$:

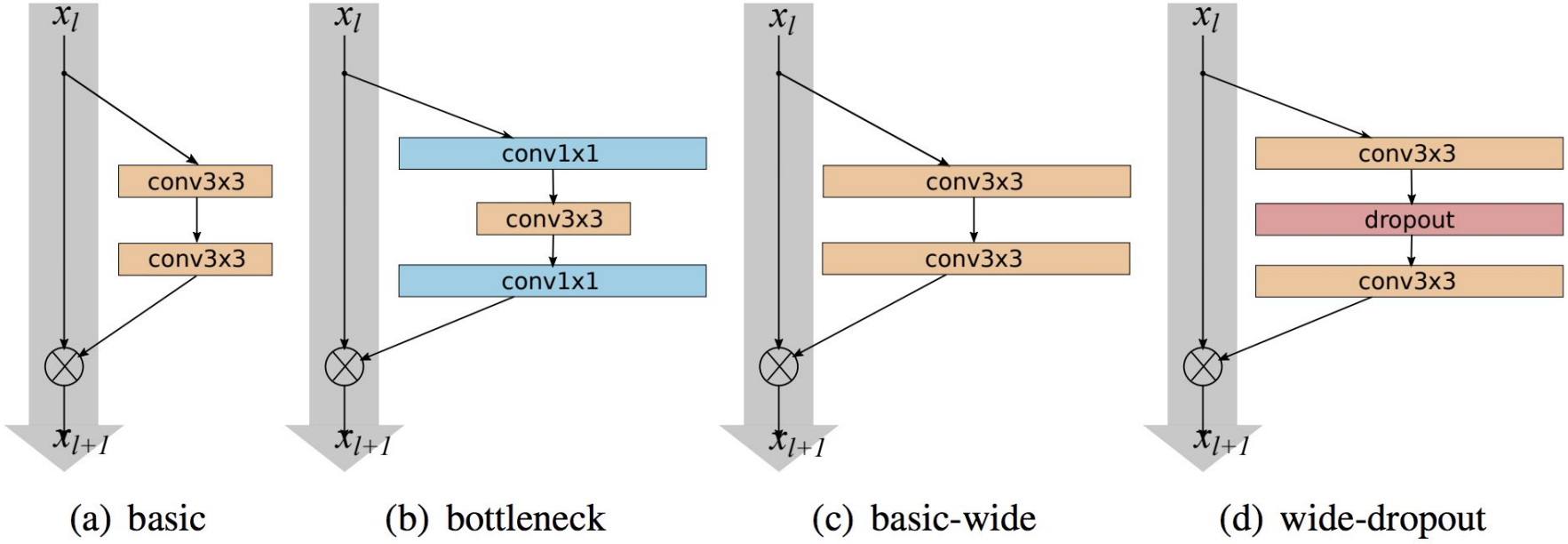
$$g^{(i)} \leftarrow \nabla_r D\left[p(y|x^{(i)}, \hat{\theta}), p(y|x^{(i)} + r, \hat{\theta})\right]\Big|_{r=\xi d^{(i)}},$$

$$r^{(i)}_{\text{vadv}} \leftarrow g^{(i)} / \|g^{(i)}\|_2$$

4) **Return**

$$\nabla_\theta \left(\frac{1}{M} \sum_{i=1}^{M} D\left[p(y|x^{(i)}, \hat{\theta}), p(y|x^{(i)} + r^{(i)}_{\text{vadv}}, \theta)\right] \right)\Bigg|_{\theta=\hat{\theta}}$$

Wide ResNet



Wide Residual Networks

Comparison

Dataset	# Labels	Supervised	Π-Model	Mean Teacher	VAT	VAT + EntMin	Pseudo-Label
CIFAR-10	4000	20.26 ± .38%	16.37 ± .63%	15.87 ± .28%	13.86 ± .27%	13.13 ± .39%	17.78 ± .57%
SVHN	1000	12.83 ± .47%	7.19 ± .27%	5.65 ± .47%	5.63 ± .20%	5.35 ± .19%	7.62 ± .29%

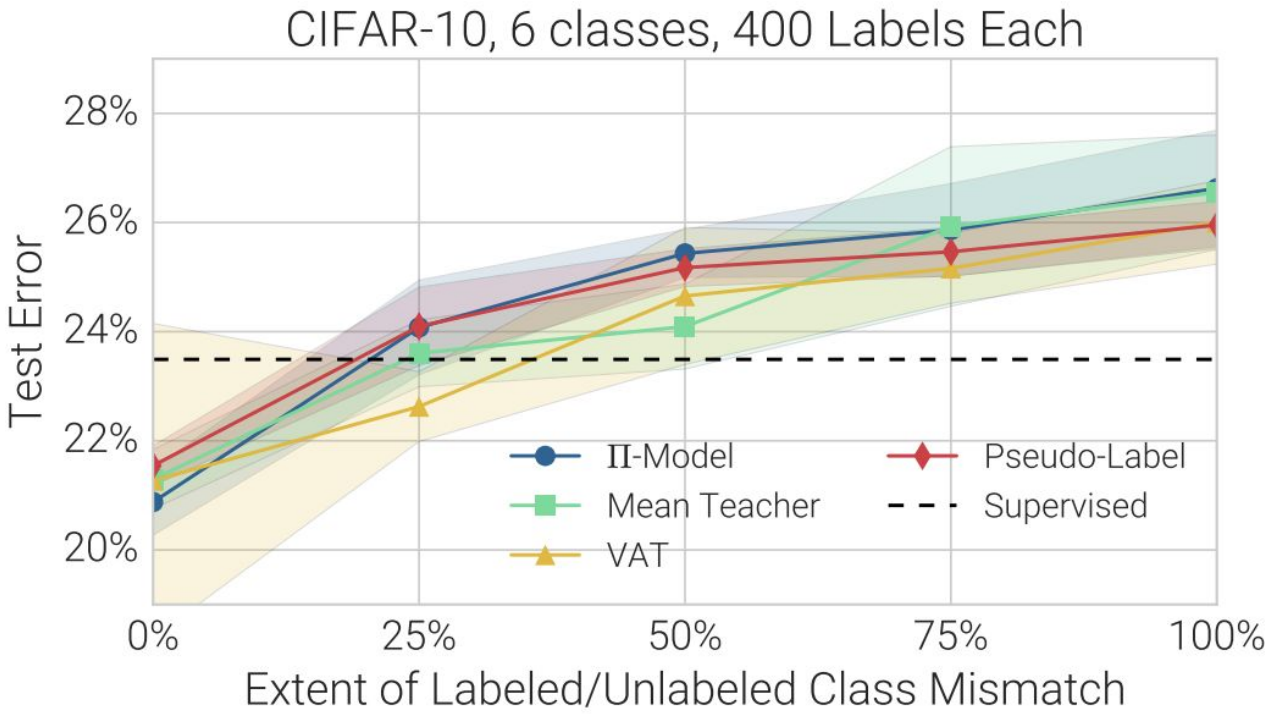
Comparison

Method	CIFAR-10 4000 Labels	SVHN 1000 Labels
Π-Model [32]	34.85% → 12.36%	19.30% → 4.80%
Π-Model [46]	13.60% → 11.29%	–
Π-Model (ours)	20.26% → 16.37%	12.83% → 7.19%
Mean Teacher [50]	20.66% → 12.31%	12.32% → 3.95%
Mean Teacher (ours)	20.26% → 15.87%	12.83% → 5.65%

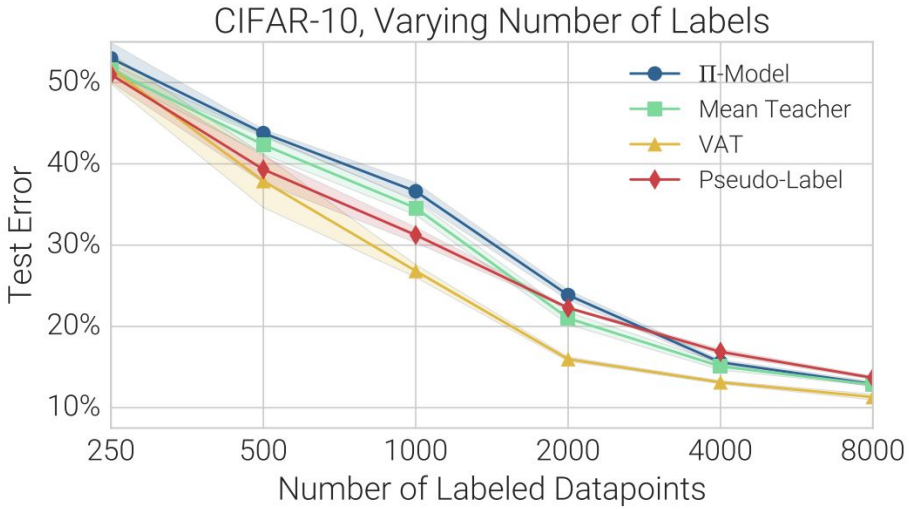
Comparison

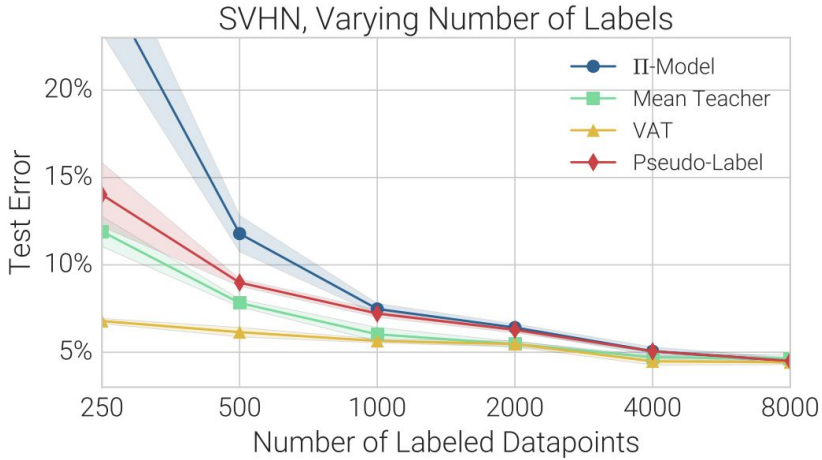
Method	CIFAR-10 4000 Labels
VAT with Entropy Minimization	13.13%
ImageNet $\rightarrow$ CIFAR-10	12.09%
ImageNet $\rightarrow$ CIFAR-10 (no overlap)	12.91%

Class Distribution Mismatch



Varying number of labels





Lessons

- Standardized architecture + equal budget for tuning hyperparameters
- Unlabeled data from a different class distribution not that useful
- Most methods don't work well in the very low labeled-data regime
- Transferring Pre-Trained Imagenet produces lower error rate
- Conclusions based on small datasets though

Agenda

- Unsupervised Data Augmentation for Consistency Training
- MixMatch: Holistic Approach to Semi-Supervised Learning
- Noisy Student (Self-Training at Scale)

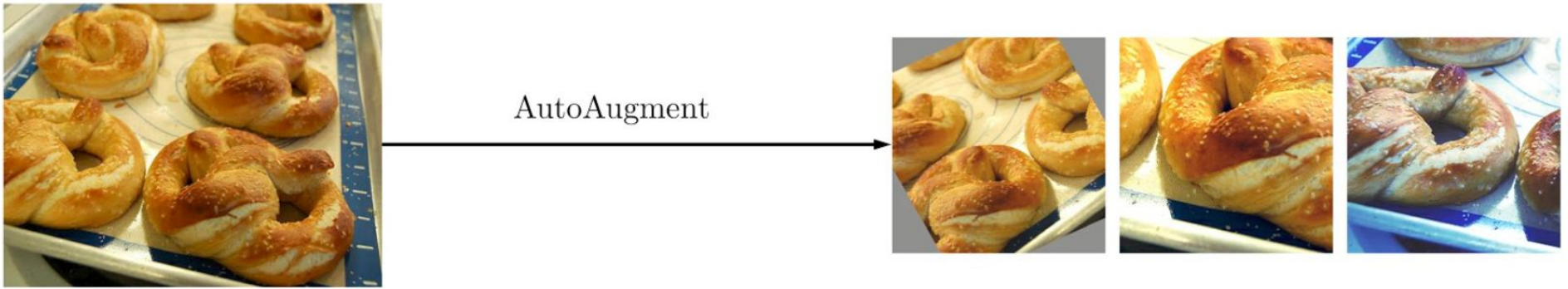
Unsupervised Data Augmentation

UNSUPERVISED DATA AUGMENTATION FOR CONSISTENCY TRAINING

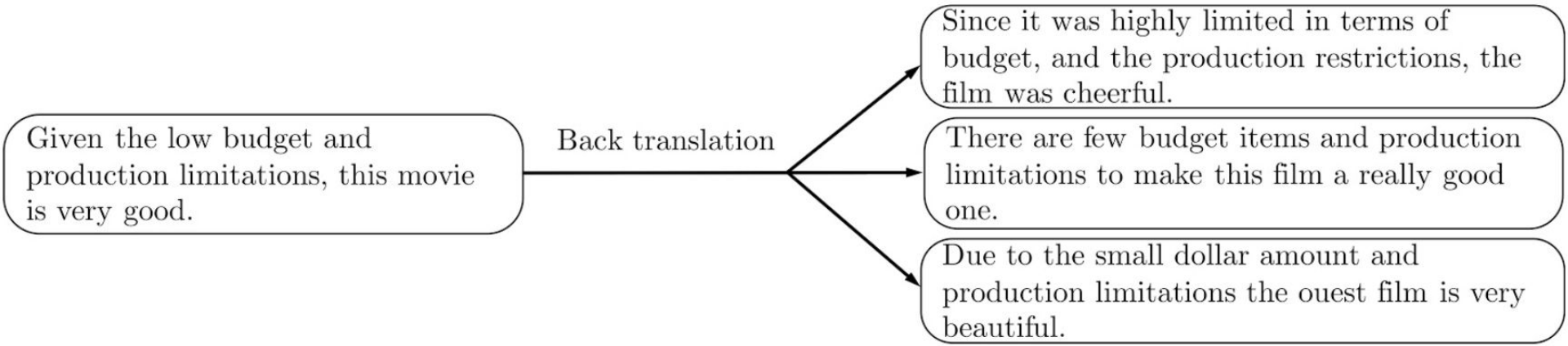
Qizhe Xie[1,2], **Zihang Dai**[1,2], **Eduard Hovy**[2], **Minh-Thang Luong**[1], **Quoc V. Le**[1]
[1] Google Research, Brain Team, [2] Carnegie Mellon University
{qizhex, dzihang, hovy}@cs.cmu.edu, {thangluong, qvl}@google.com

Slides / Figures from Thang Luong

Unsupervised Data Augmentation



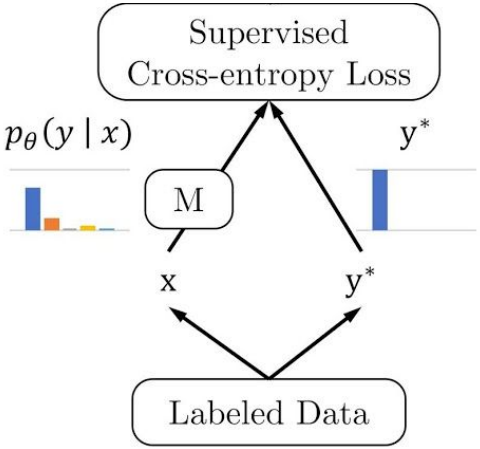
Unsupervised Data Augmentation



Unsupervised Data Augmentation

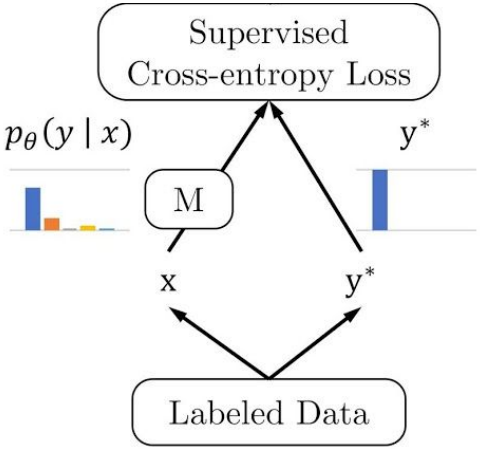
Key Idea: Apply SOTA Data Augmentation to unlabeled data via consistency training in semi-supervised learning

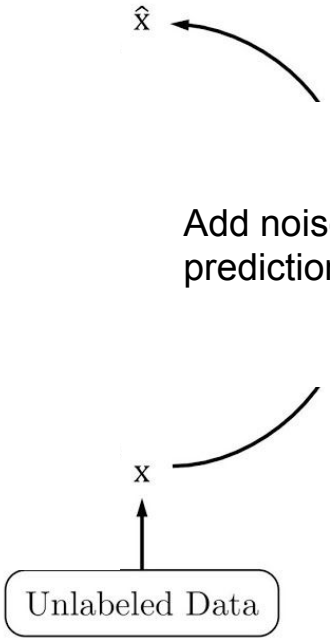
Consistency Training in Semi-Supervised Learning



Unlabeled Data

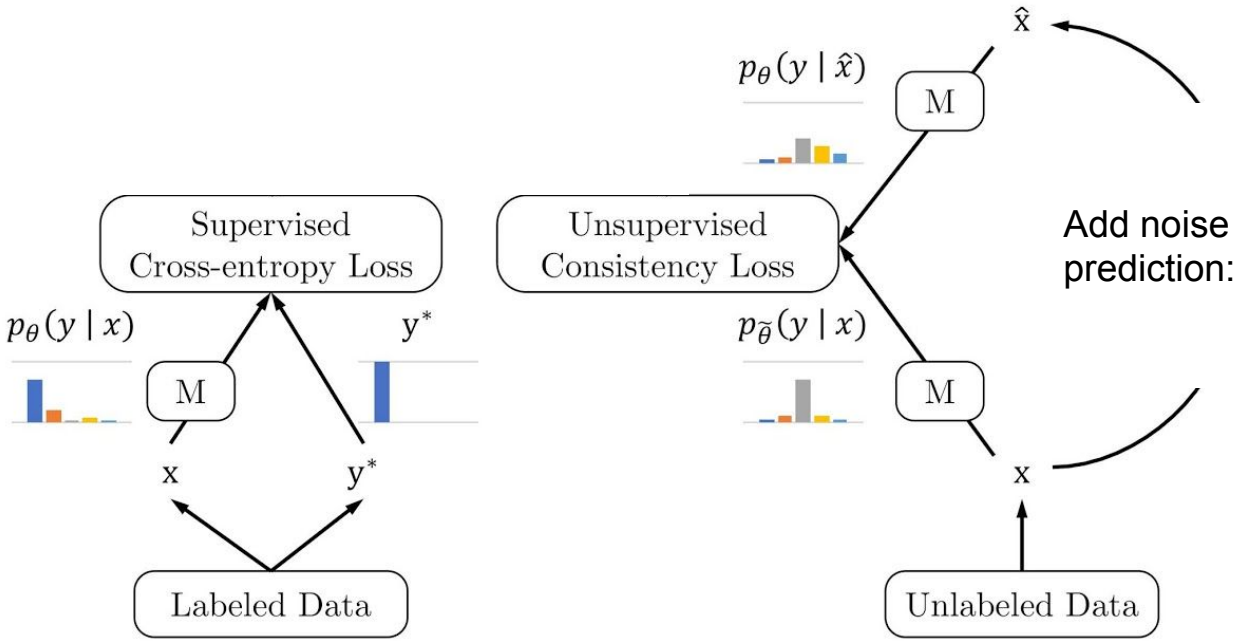
Consistency Training in Semi-Supervised Learning





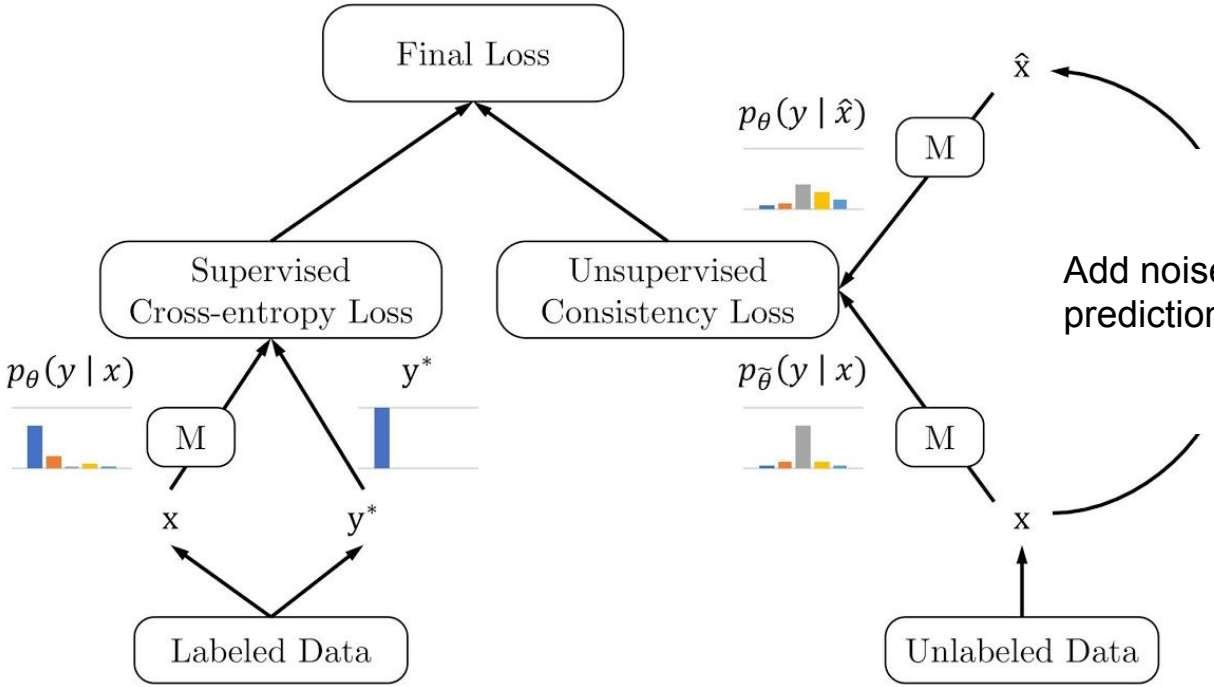
Add noise to regularize model prediction: VAT [Miyato et al., 2018]

Consistency Training in Semi-Supervised Learning



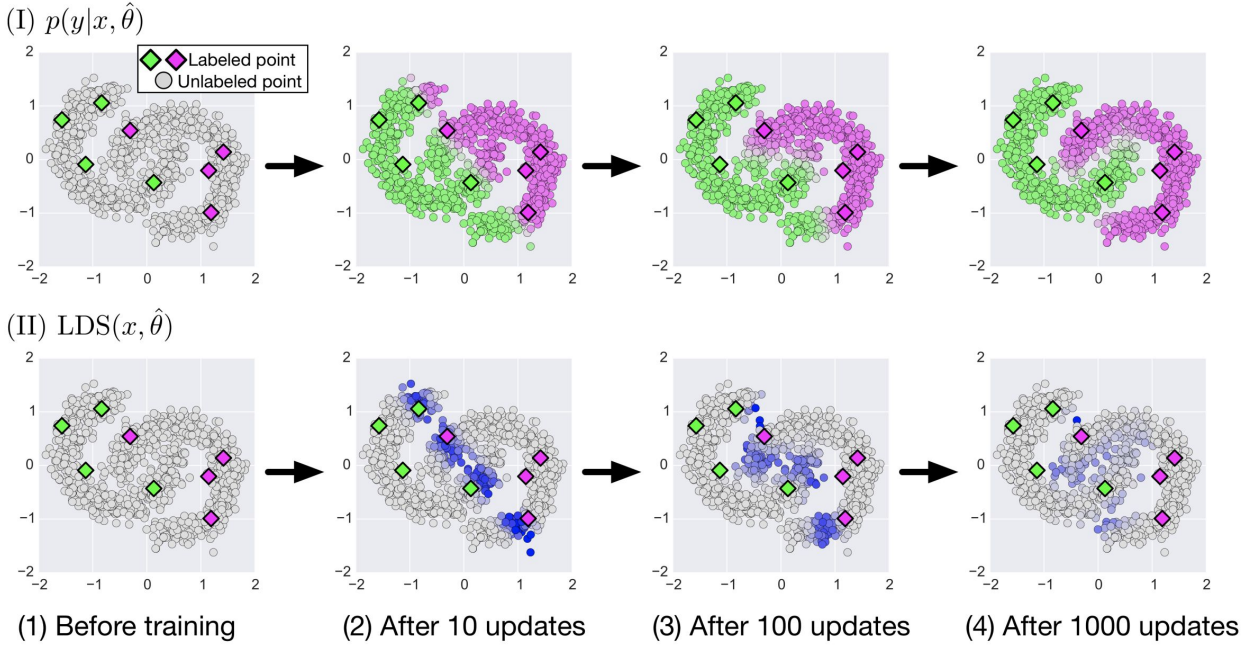
Add noise to regularize model prediction: VAT [Miyato et al., 2018]

Consistency Training in Semi-Supervised Learning



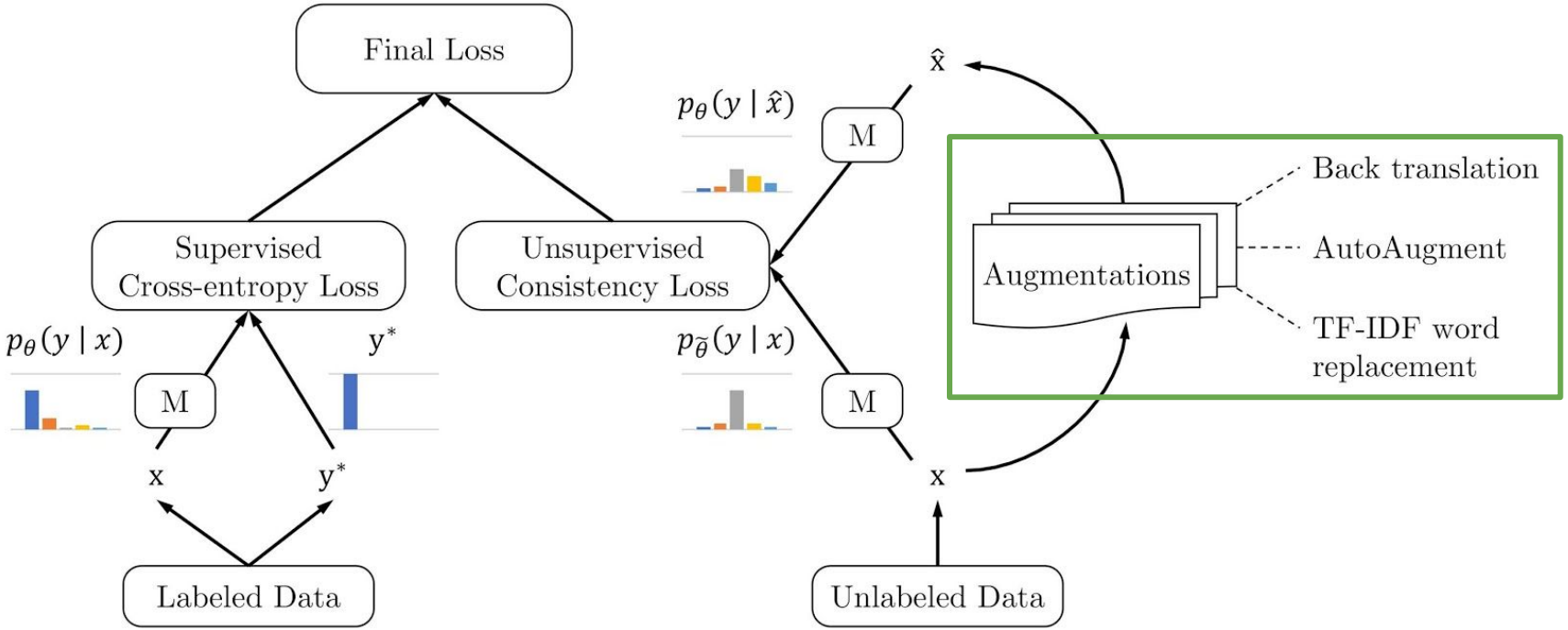
Add noise to regularize model prediction: VAT [Miyato et al., 2018]

Label propagation

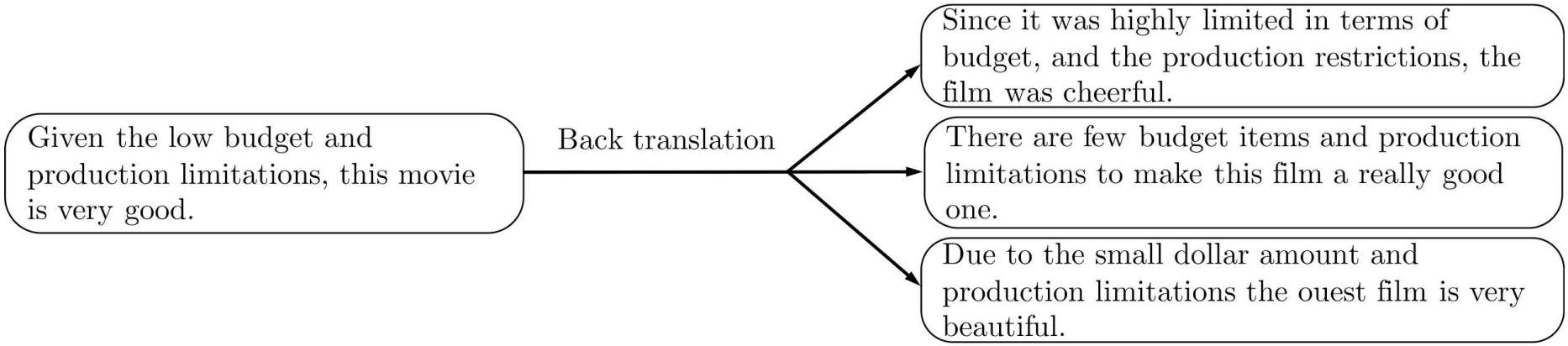


Graph taken from VAT (Miyato et al. 2017)

Unsupervised Data Augmentation (UDA)

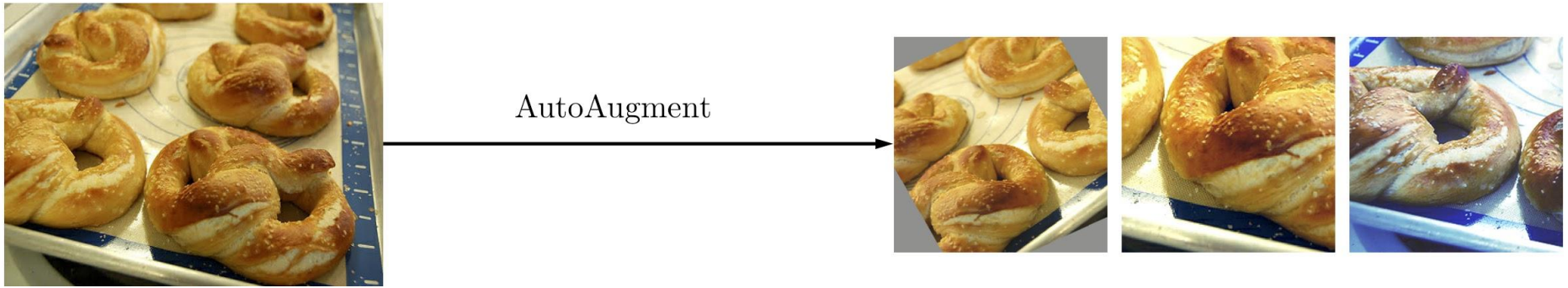


Augmentation provides Diverse and Valid Perturbations



- Back translation for Text Classification:
  - English –> French –> English
  - Sampling: diverse (high-temperature) vs valid (low-temperature).

Augmentation injects task-specific knowledge



- AutoAugment for Image Classification:
  - Example policies: (Rotate, 0.8, 2), (Brightness, 0.8, 4)

Language Experiments

Text Classification

Fully supervised baseline						
Datasets (# Sup examples)	IMDb (25k)	Yelp-2 (560k)	Yelp-5 (650k)	Amazon-2 (3.6m)	Amazon-5 (3m)	DBpedia (560k)
Pre-BERT SOTA	*4.32*	2.16	29.98	3.32	34.81	0.70
$\text{BERT}_{\text{LARGE}}$	4.51	*1.89*	*29.32*	*2.63*	*34.17*	*0.64*

Text Classification

Fully supervised baseline

Datasets (# Sup examples)	IMDb (25k)	Yelp-2 (560k)	Yelp-5 (650k)	Amazon-2 (3.6m)	Amazon-5 (3m)	DBpedia (560k)
Pre-BERT SOTA	*4.32*	2.16	29.98	3.32	34.81	0.70
$\text{BERT}_{\text{LARGE}}$	4.51	*1.89*	*29.32*	*2.63*	*34.17*	*0.64*

Semi-supervised setting

Initialization	UDA	IMDb (20)	Yelp-2 (20)	Yelp-5 (2.5k)	Amazon-2 (20)	Amazon-5 (2.5k)	DBpedia (140)
Random	✗						
	✓						
$\text{BERT}_{\text{BASE}}$	✗						
	✓						
$\text{BERT}_{\text{LARGE}}$	✗						
	✓						
$\text{BERT}_{\text{FINETUNE}}$	✗						
	✓						

- Four initialization settings
- 3-4 orders magnitude less data

Text Classification

Fully supervised baseline							
Datasets (# Sup examples)		IMDb (25k)	Yelp-2 (560k)	Yelp-5 (650k)	Amazon-2 (3.6m)	Amazon-5 (3m)	DBpedia (560k)
Pre-BERT SOTA		*4.32*	2.16	29.98	3.32	34.81	0.70
$\text{BERT}_{\text{LARGE}}$		4.51	*1.89*	*29.32*	*2.63*	*34.17*	*0.64*
Semi-supervised setting							
Initialization	**UDA**	IMDb (20)	Yelp-2 (20)	Yelp-5 (2.5k)	Amazon-2 (20)	Amazon-5 (2.5k)	DBpedia (140)
Random	✗	43.27	40.25	50.80	45.39	55.70	41.14
	✓	25.23	8.33	41.35	16.16	44.19	7.24
$\text{BERT}_{\text{BASE}}$	✗	27.56	13.60	41.00	26.75	44.09	2.58
	✓	5.45	2.61	33.80	3.96	38.40	1.33
$\text{BERT}_{\text{LARGE}}$	✗	11.72	10.55	38.90	15.54	42.30	1.68
	✓	4.78	2.50	33.54	3.93	37.80	1.09
$\text{BERT}_{\text{FINETUNE}}$	✗	6.50	2.94	32.39	12.17	37.32	-
	✓	**4.20**	**2.05**	**32.08**	**3.50**	**37.12**	-

SOTA on IMDb with 20 examples!

Training Signal Annealing (TSA)

- Prevent overtraining on labeled data

Training Signal Annealing (TSA)

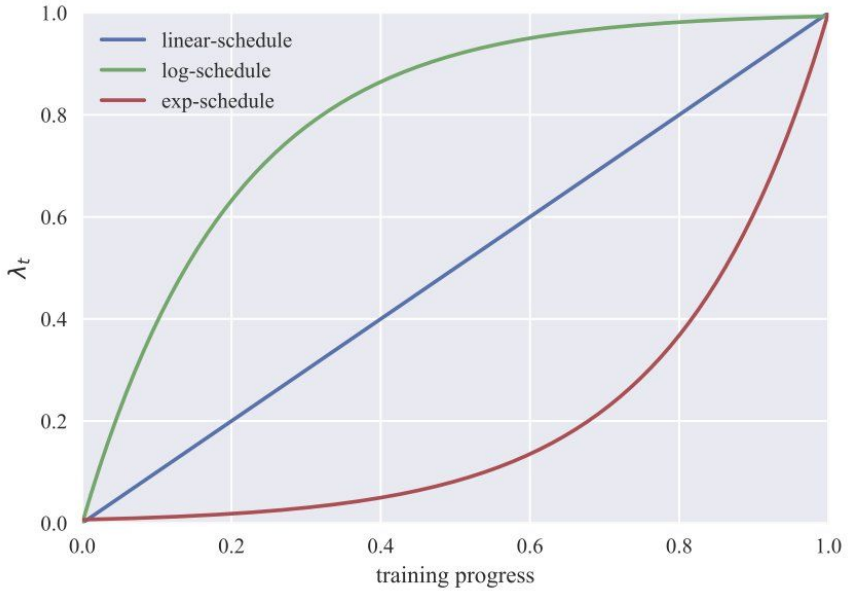
- Prevent overtraining on labeled data
- Mask out labeled examples based on confidence scores

$$\min_{\theta} \frac{1}{Z} \sum_{x,y \in B} \left[-\log p_\theta(y \mid x) \, I(p_\theta(y \mid x) < \eta_t) \right]$$

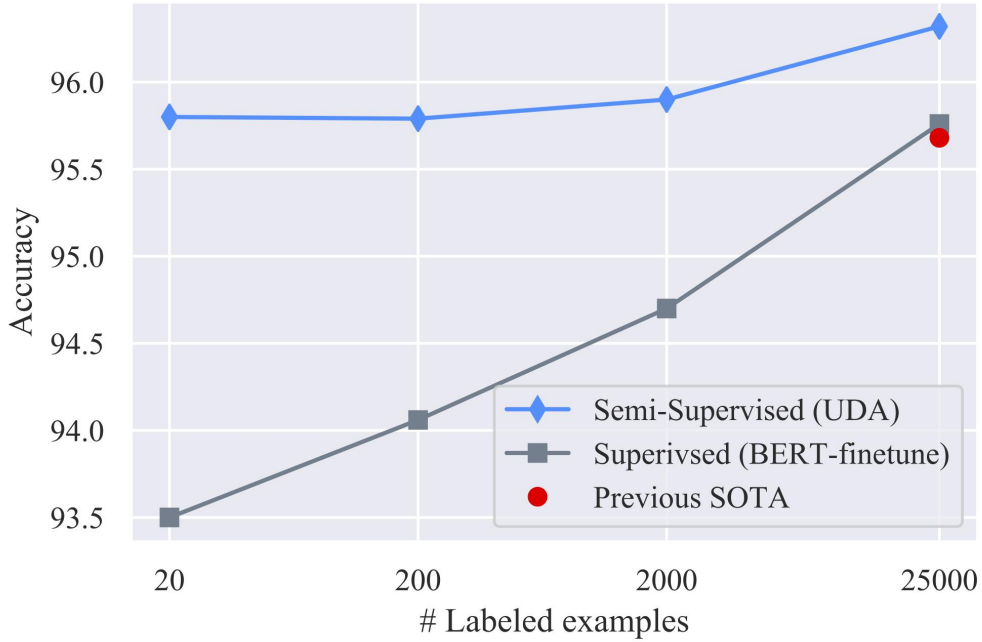
Training Signal Annealing (TSA)

- Prevent overtraining on labeled data
- Mask out labeled examples based on confidence scores

$$\eta_t = \frac{1}{k} + \lambda_t * (1 - \frac{1}{k})$$



IMDb Results with Different Labeled Data Sizes



Vision Experiments

SSL Benchmarks on CIFAR-10 and SVHN

Methods	Model	# Param	CIFAR-10 (4k)	SVHN (1k)
Pseudo-Label	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
mixmixup	WRN-28-2	1.5M	10	-
ICT	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
Π-Model	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
VAdD	Conv-Large	3.1M	11.32 ± 0.11	4.16 ± 0.08
Fast-SWA	Conv-Large	3.1M	9.05	-
ICT	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Mean Teacher	Shake-Shake	26M	6.28 ± 0.15	-
Fast-SWA	Shake-Shake	26M	5.0	-

SSL Benchmarks on CIFAR-10 and SVHN

Methods	Model	# Param	CIFAR-10 (4k)	SVHN (1k)
Pseudo-Label	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
mixmixup	WRN-28-2	1.5M	10	-
ICT	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
Π-Model	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
VAdD	Conv-Large	3.1M	11.32 ± 0.11	4.16 ± 0.08
Fast-SWA	Conv-Large	3.1M	9.05	-
ICT	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Mean Teacher	Shake-Shake	26M	6.28 ± 0.15	-
Fast-SWA	Shake-Shake	26M	5.0	-
UDA	WRN-28-2	1.5M	5.27 ± 0.11	$\mathbf{2.46 \pm 0.17}$
UDA	Shake-Shake	26M	3.6	-
UDA	PyramidNet+ShakeDrop	26M	**2.7**	-

30% error reduction from the previous SOTA with WRN-28-2.

SSL Benchmarks on CIFAR-10 and SVHN

Methods	Model	# Param	CIFAR-10 (4k)	SVHN (1k)
Pseudo-Label	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
mixmixup	WRN-28-2	1.5M	10	-
ICT	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
Π-Model	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
VAdD	Conv-Large	3.1M	11.32 ± 0.11	4.16 ± 0.08
Fast-SWA	Conv-Large	3.1M	9.05	-
ICT	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Mean Teacher	Shake-Shake	26M	6.28 ± 0.15	-
Fast-SWA	Shake-Shake	26M	5.0	-
UDA	WRN-28-2	1.5M	5.27 ± 0.11	$\mathbf{2.46 \pm 0.17}$
UDA	Shake-Shake	26M	3.6	-
UDA	PyramidNet+ShakeDrop	26M	**2.7**	-

Further advancing the SOTA with larger networks.

Results with more advanced architectures

Methods	# Sup	Wide-ResNet-28-2	Shake-Shake	ShakeDrop
Supervised	50k	5.4	2.9	2.7
AutoAugment		4.3	2.0	1.5
UDA	4k	5.3	3.6	2.7

With only 4k labeled examples, UDA nearly matches the fully supervised results

Ablation study on data augmentation

Augmentation	CIFAR-10	SVHN
Cropping & flipping	16.17	8.27
Cutout	6.42	3.09
Switched Augment	5.59	2.74
AutoAugment	**5.10**	**2.22**

State-of-the-art augmentation is important!

Summary

- Data augmentation is an effective perturbation for SSL.
- UDA significantly improves for both language and vision.
- UDA combines well with transfer learning, e.g., BERT.

ImageNet 10% Labeled Examples Experiments

10% labeled data (ResNet-50)

Methods	top-1 acc	top-5 acc
Supervised	55.09	77.26
UDA	**68.66**	**88.52**

Unlabeled data:
ImageNet

ImageNet Full Data Experiments

- Obtaining extra in-domain unlabeled data:
  - Directly using out-of-domain unlabeled data hurts performance
  - Use baseline to label out-of-domain images to infer the labels for out-of-domain data
  - For each category, obtain the ones with the highest probabilities

Full labeled data (ResNet-50)

Methods	top-1 acc	top-5 acc
Supervised	77.28	93.73
AutoAugment	78.28	94.36
UDA	**79.04**	**94.45**

Unlabeled data:

1.3M images from JFT

Ablation study on TSA

- Yelp-5:
  - A lot of unlabeled examples
  - Limited number of labeled examples
  - Exp-schedule is the best
- CIFAR-10:
  - 4,000 labeled examples
  - 50,000 labeled examples
  - Linear-schedule is the best

TSA schedule	Yelp-5	CIFAR-10
✗	50.81	5.67
log-schedule	49.06	5.41
linear-schedule	45.41	**5.10**
exp-schedule	**41.35**	7.25

Additional Techniques

- Techniques to sharpen predictions:
  - Entropy minimization
  - Softmax temperature controlling
  - Confidence-based masking
- Domain-relevance data filtering: filter in-domain unlabeled data from out-of-domain unlabeled data

MixMatch

MixMatch: A Holistic Approach to Semi-Supervised Learning

David Berthelot
Google Research
dberth@google.com

Nicholas Carlini
Google Research
ncarlini@google.com

Ian Goodfellow
Work done at Google
ian-academic@mailfence.com

Avital Oliver
Google Research
avitalo@google.com

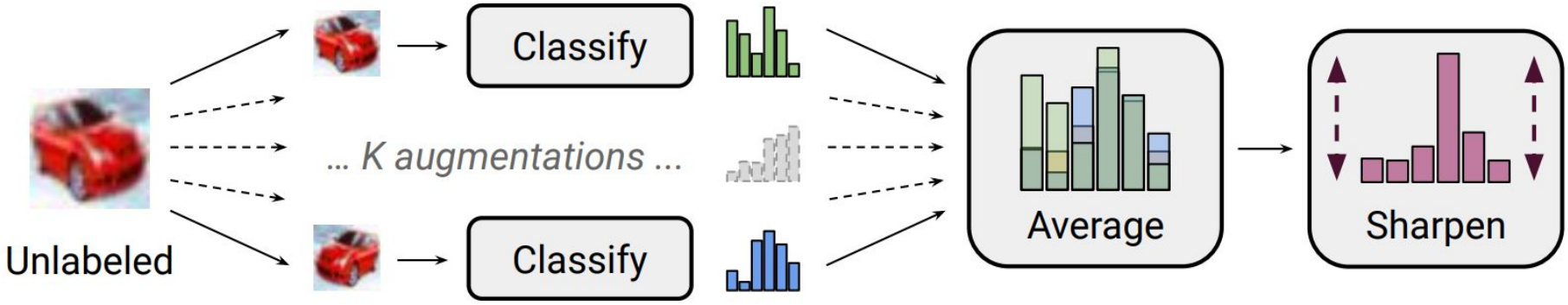
Nicolas Papernot
Google Research
papernot@google.com

Colin Raffel
Google Research
craffel@google.com

Abstract

Semi-supervised learning has proven to be a powerful paradigm for leveraging unlabeled data to mitigate the reliance on large labeled datasets. In this work, we unify the current dominant approaches for semi-supervised learning to produce a new algorithm, MixMatch, that guesses low-entropy labels for data-augmented unlabeled examples and mixes labeled and unlabeled data using MixUp. MixMatch obtains state-of-the-art results by a large margin across many datasets and labeled data amounts. For example, on CIFAR-10 with 250 labels, we reduce error rate by a factor of 4 (from 38% to 11%) and by a factor of 2 on STL-10. We also demonstrate how MixMatch can help achieve a dramatically better accuracy-privacy trade-off for differential privacy. Finally, we perform an ablation study to tease apart which components of MixMatch are most important for its success. We release all code used in our experiments.[1]

MixMatch


Classify
... K augmentations ...
Classify
Unlabeled
Average
Sharpen

MixMatch

$$\lambda \sim \text{Beta}(\alpha, \alpha)$$
$$\lambda' = \max(\lambda, 1 - \lambda)$$
$$x' = \lambda' x_1 + (1 - \lambda') x_2$$
$$p' = \lambda' p_1 + (1 - \lambda') p_2$$

MixUp

MixMatch

Algorithm 1 MixMatch takes a batch of labeled data $\mathcal{X}$ and a batch of unlabeled data $\mathcal{U}$ and produces a collection $\mathcal{X}'$ (resp. $\mathcal{U}'$) of processed labeled examples (resp. unlabeled with guessed labels).

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \big((x_b, p_b); b \in (1, \dots, B)\big)$, batch of unlabeled examples $\mathcal{U} = \big(u_b; b \in (1, \dots, B)\big)$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.

for $b = 1$ **to** B **do**

  $\hat{x}_b = \text{Augment}(x_b)$ *// Apply data augmentation to x_b*

  for $k = 1$ **to** K **do**

    $\hat{u}_{b,k} = \text{Augment}(u_b)$ *// Apply k^{th} round of data augmentation to u_b*

  end for

  $\bar{q}_b = \frac{1}{K} \sum_k \text{p}_{\text{model}}(y \mid \hat{u}_{b,k}; \theta)$ *// Compute average predictions across all augmentations of u_b*

  $q_b = \text{Sharpen}(\bar{q}_b, T)$ *// Apply temperature sharpening to the average prediction (see eq. (7))*

end for

$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \dots, B)\big)$ *// Augmented labeled examples and their labels*

$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \dots, B), k \in (1, \dots, K)\big)$ *// Augmented unlabeled examples, guessed labels*

$\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ *// Combine and shuffle labeled and unlabeled data*

$\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \dots, |\hat{\mathcal{X}}|)\big)$ *// Apply MixUp to labeled data and entries from $\mathcal{W}$*

$\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \dots, |\hat{\mathcal{U}}|)\big)$ *// Apply MixUp to unlabeled data and the rest of $\mathcal{W}$*

return $\mathcal{X}', \mathcal{U}'$

MixMatch

$$\mathcal{X}', \mathcal{U}' = \text{MixMatch}(\mathcal{X}, \mathcal{U}, T, K, \alpha)$$

$$\mathcal{L}_{\mathcal{X}} = \frac{1}{|\mathcal{X}'|} \sum_{x,p \in \mathcal{X}'} \text{H}(p, \text{p}_{\text{model}}(y \mid x; \theta))$$

$$\mathcal{L}_{\mathcal{U}} = \frac{1}{L|\mathcal{U}'|} \sum_{u,q \in \mathcal{U}'} \|q - \text{p}_{\text{model}}(y \mid u; \theta)\|_2^2$$

$$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_{\mathcal{U}} \mathcal{L}_{\mathcal{U}}$$

MixMatch

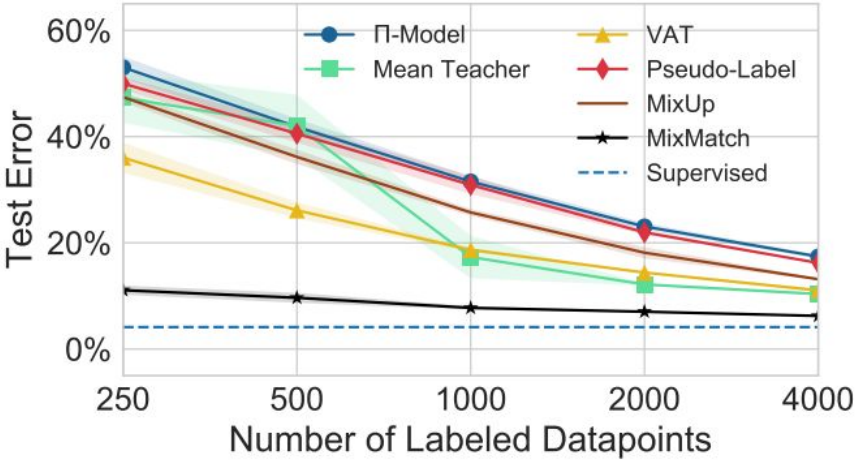


Figure 2: Error rate comparison of MixMatch to baseline methods on CIFAR-10 for a varying

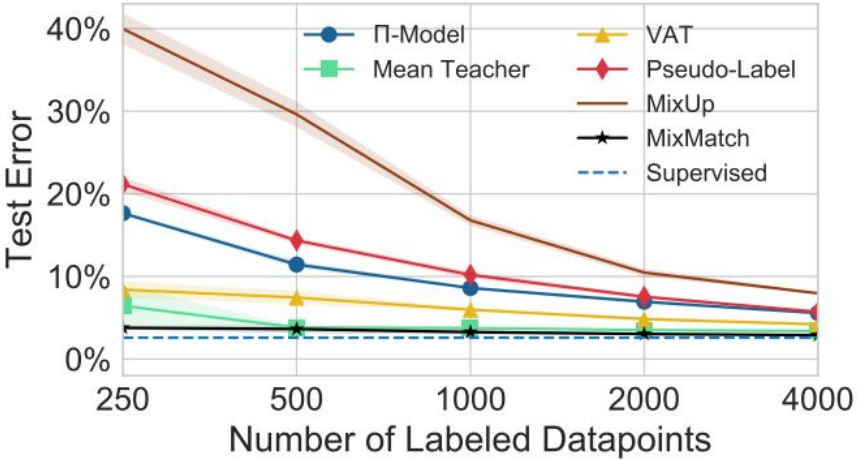


Figure 3: Error rate comparison of MixMatch to baseline methods on SVHN for a varying num-

MixMatch

Methods/Labels	250	500	1000	2000	4000
PiModel	53.02 ± 2.05	41.82 ± 1.52	31.53 ± 0.98	23.07 ± 0.66	17.41 ± 0.37
PseudoLabel	49.98 ± 1.17	40.55 ± 1.70	30.91 ± 1.73	21.96 ± 0.42	16.21 ± 0.11
Mixup	47.43 ± 0.92	36.17 ± 1.36	25.72 ± 0.66	18.14 ± 1.06	13.15 ± 0.20
VAT	36.03 ± 2.82	26.11 ± 1.52	18.68 ± 0.40	14.40 ± 0.15	11.05 ± 0.31
MeanTeacher	47.32 ± 4.71	42.01 ± 5.86	17.32 ± 4.00	12.17 ± 0.22	10.36 ± 0.25
MixMatch	11.08 ± 0.87	9.65 ± 0.94	7.75 ± 0.32	7.03 ± 0.15	6.24 ± 0.06

Table 5: Error rate (%) for CIFAR10.

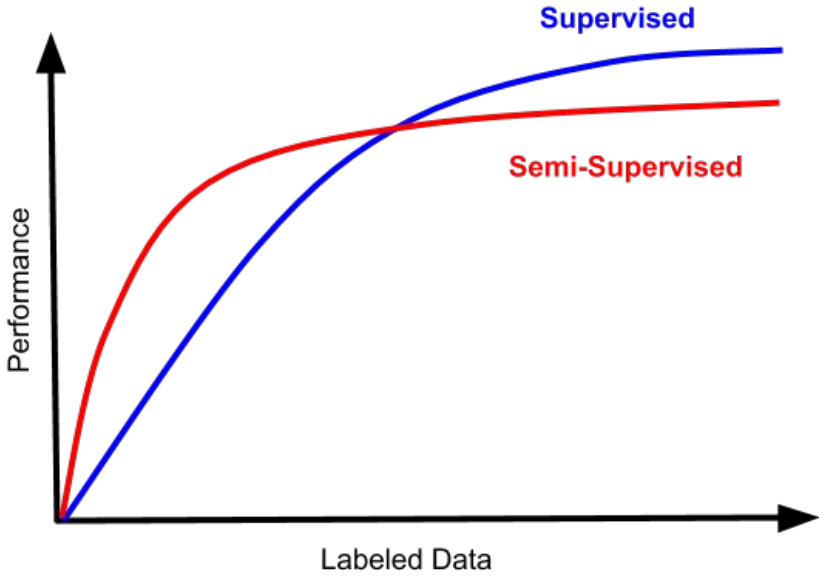
Noisy Student

Self-training with Noisy Student improves ImageNet classification

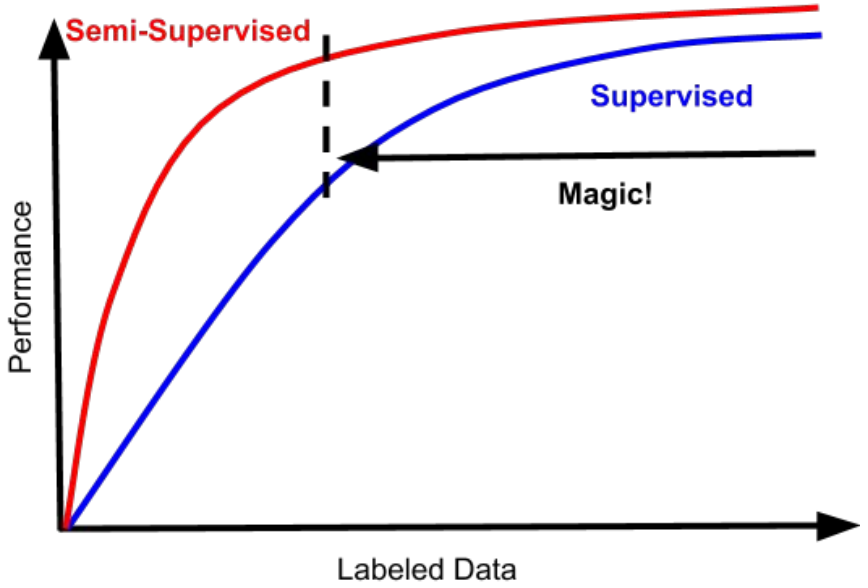
Qizhe Xie* [1], Minh-Thang Luong[1], Eduard Hovy[2], Quoc V. Le[1]
[1]Google Research, Brain Team, [2]Carnegie Mellon University
{qizhex, thangluong, qvl}@google.com, hovy@cmu.edu

Slides / Figures from Thang Luong

Background: Semi-supervised Learning (SSL)



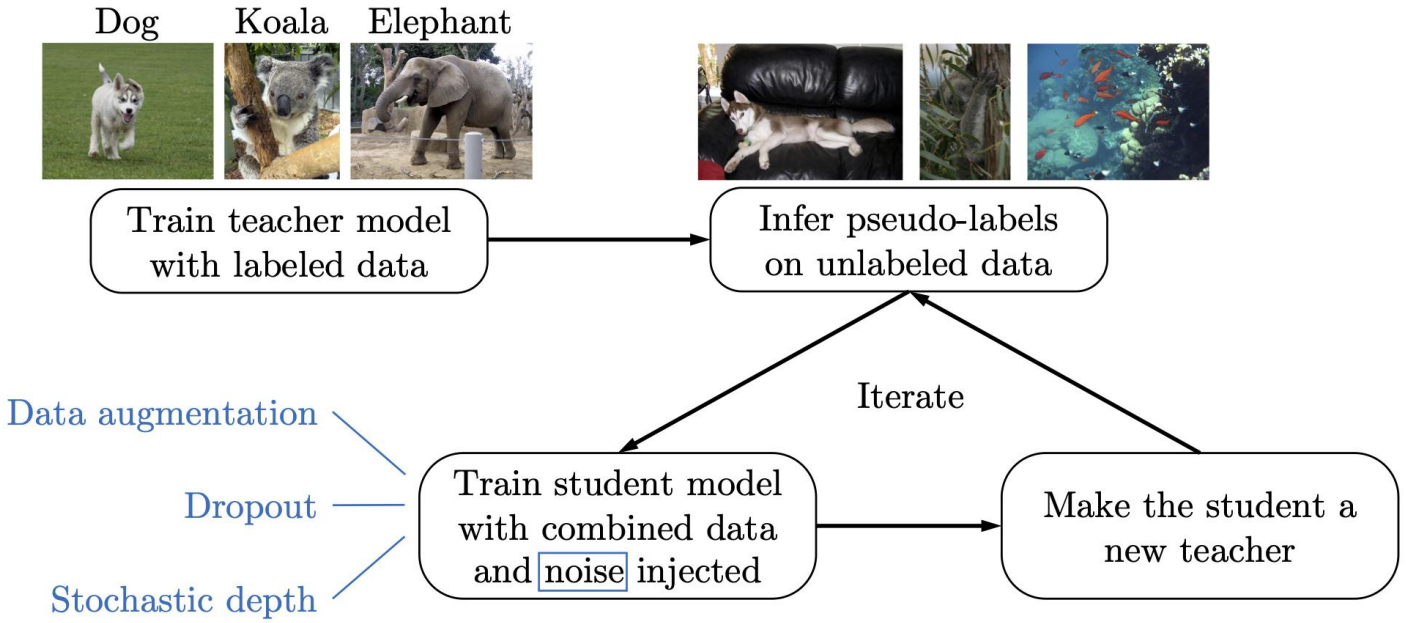
Belief of many ML practitioners



Belief of many SSL researchers

Figure credit: Vincent Vanhoucke

Self-training with Noisy Student



Experiment Settings

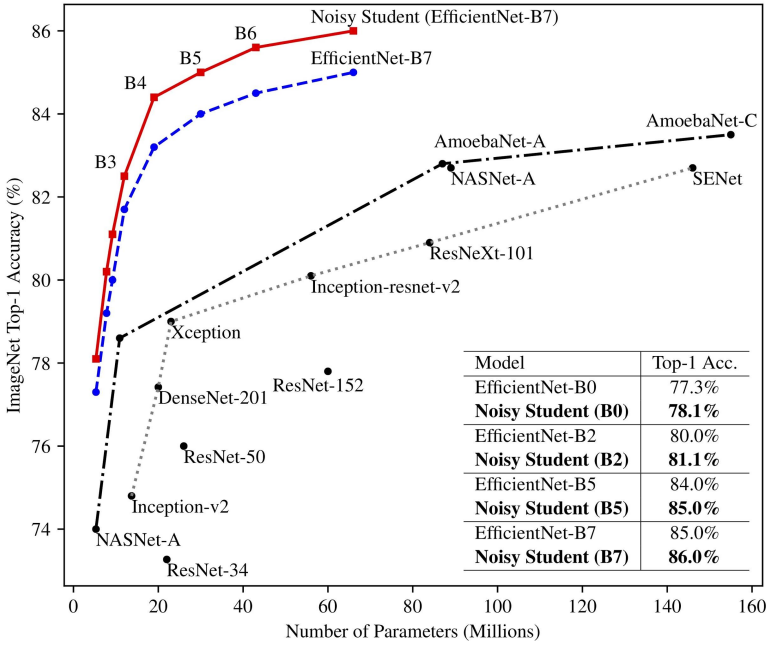
- Noise:
  - Input noise: RandAugment data augmentation.
  - Model noise: Dropout, Stochastic depth.
- Architecture: EfficientNets.
- Pseudo-labels: soft pseudo-labels (continuous).
- Labeled dataset: ImageNet (1.3M images).
- Unlabeled dataset: JFT (300M unlabeled images).
- Iterative training: B7->L2->L2->L2

ImageNet Results

Method	# Param	Extra Data	Top-1 Acc.	Top-5 Acc.
GPipe	557M	-	84.3%	97.0%
EfficientNet-B7	66M	-	85.0%	97.2%
EfficientNet-L2	480M	-	85.5%	97.5%
ResNeXt-101 WSL	829M	3.5B instagram images labeled with tags	85.4%	97.6%
FixRes ResNeXt-101 WSL	829M	3.5B instagram images labeled with tags	86.4%	98.0%
Noisy Student (EfficientNet-L2)	480M	300M unlabeled images	**88.4%**	**98.7%**

- 2% improvement of top-1 accuracy.
- 35% error reduction of top-5 accuracy.
- One order of magnitude less data without labels.
- Twice as small in the number of parameters.

Performance on small models without iterative training



Model	Top-1 Acc.
EfficientNet-B0	77.3%
Noisy Student (B0)	**78.1%**
EfficientNet-B2	80.0%
Noisy Student (B2)	**81.1%**
EfficientNet-B5	84.0%
Noisy Student (B5)	**85.0%**
EfficientNet-B7	85.0%
Noisy Student (B7)	**86.0%**

- The same model is used as the teacher and the student.
- Much better tradeoff in terms of accuracy and model size.

Robustness results on ImageNet-A

Method	Top-1 Acc.	Top-5 Acc.
ResNet-101 [30]	4.7%	-
ResNeXt-101 [30] (32x4d)	5.9%	-
ResNet-152 [30]	6.1%	-
ResNeXt-101 [30] (64x4d)	7.3%	-
DPN-98 [30]	9.4%	-
ResNeXt-101+SE [30] (32x4d)	14.2%	-
ResNeXt-101 WSL [51, 55]	61.0%	-
EfficientNet-L2	49.6%	78.6%
Noisy Student (L2)	**83.7%**	**95.2%**

ImageNet-A: difficult images that cause significant drops in accuracy to state-of-the-art models.

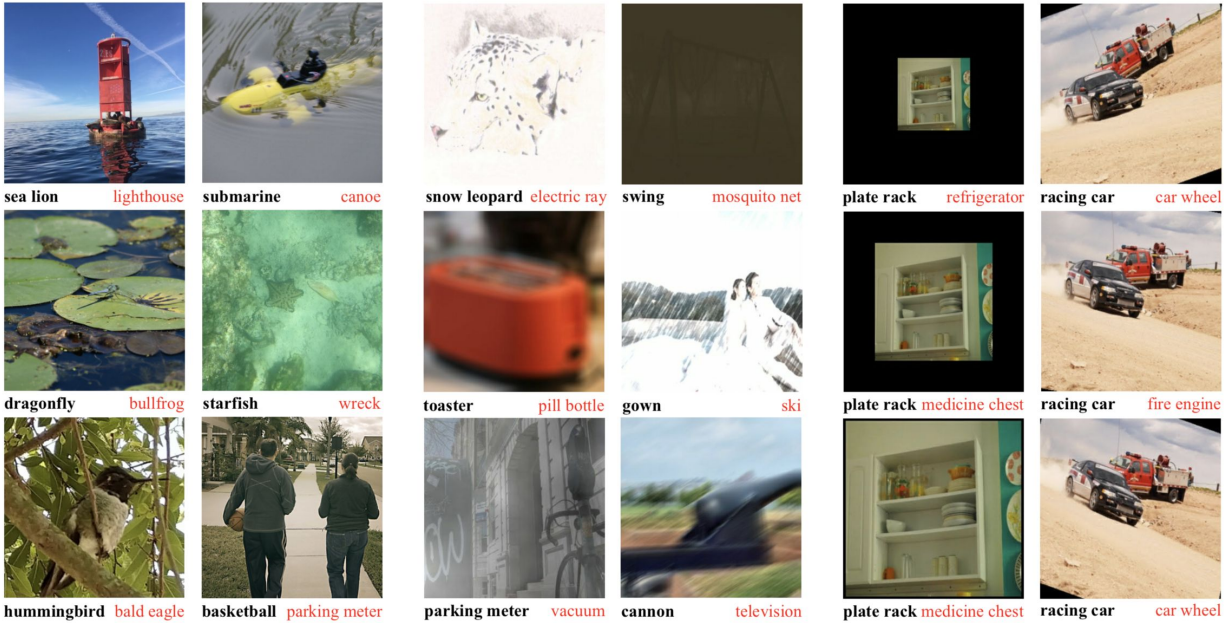
Robustness results on ImageNet-C, P

Method	Res.	Top-1 Acc.	mCE
ResNet-50 [29]	224	39.0%	76.7
SIN [22]	224	45.2%	69.3
Patch Gaussian [47]	299	52.3%	60.4
ResNeXt-101 WSL [51, 55]	224	-	45.7
EfficientNet-L2	224	62.6%	47.5
Noisy Student (L2)	224	76.5%	30.0
EfficientNet-L2	299	66.6%	42.5
Noisy Student (L2)	299	**77.8%**	**28.3**

Method	Res.	Top-1 Acc.	mFR
ResNet-50 [29]	224	-	58.0
Low Pass Filter Pooling [92]	224	-	51.2
ResNeXt-101 WSL [51, 55]	224	-	27.8
EfficientNet-L2	224	80.4%	27.2
Noisy Student (L2)	224	85.2%	14.2
EfficientNet-L2	299	81.6%	23.7
Noisy Student (L2)	299	**86.4%**	**12.2**

- ImageNet-C and P: images with common corruptions and perturbations such as blurring, fogging, rotation and scaling.
- Lower is better for mean corruption error (mCE) and mean flip rate (mFR).

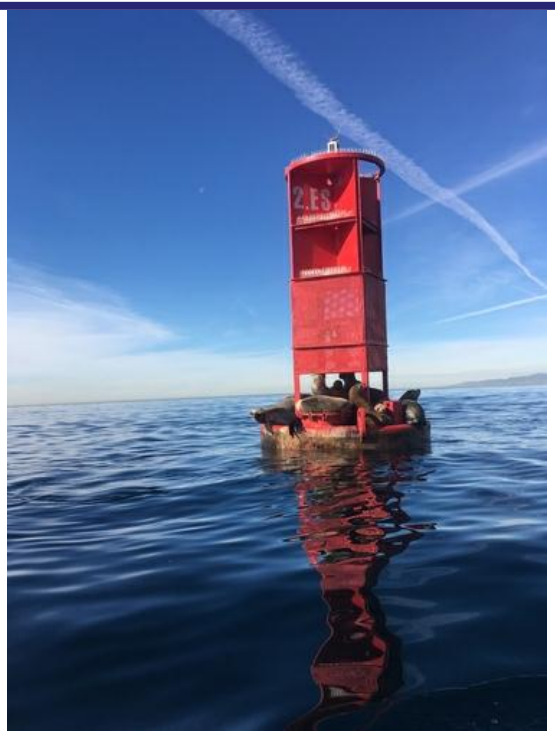
Qualitative Analysis on ImageNet-A, C and P



(a) ImageNet-A (b) ImageNet-C (c) ImageNet-P

Black texts are correct predictions made by our model and red texts are incorrect predictions by our baseline model.

Qualitative Analysis on ImageNet-A



sea lion lighthouse

dragonfly bull frog

Qualitative Analysis on ImageNet-C

parking meter vacuum

swing mosquito net

The Importance of Noise in Self-training

Why can the student model outperform the teacher model?

Model / Unlabeled Set Size	1.3M	130M
EfficientNet-B5	83.3%	84.0%
Noisy Student (B5)	**83.9%**	**84.9%**
student w/o Aug	83.6%	84.6%
student w/o Aug, SD, Dropout	83.2%	84.3%
teacher w. Aug, SD, Dropout	83.7%	84.4%

- Standard data augmentation is used when we use 1.3M unlabeled images.
- RandAugment is used when we use 130M unlabeled images.

Summary of Semi-Supervised Learning

- Semi-Supervised Learning is a practically important problem for two scenarios:
  - Lot of labeled data and a lot more unlabeled data
  - Very little labeled data and plenty of unlabeled data

- Promise early on and lately excellent results for the second scenario

- Not much for the first scenario until recently
  - Noisy Student

Outline

- Intro
- Semi-Supervised Learning
- ***Unsupervised Distribution Alignment***

Distribution Alignment Problem

- Image to Image

(Isola et al, 2017)

Distribution Alignment Problem

- Image to Image

(Isola et al, 2017)

Distribution Alignment Problem

- Image to Image

(Isola et al, 2017)

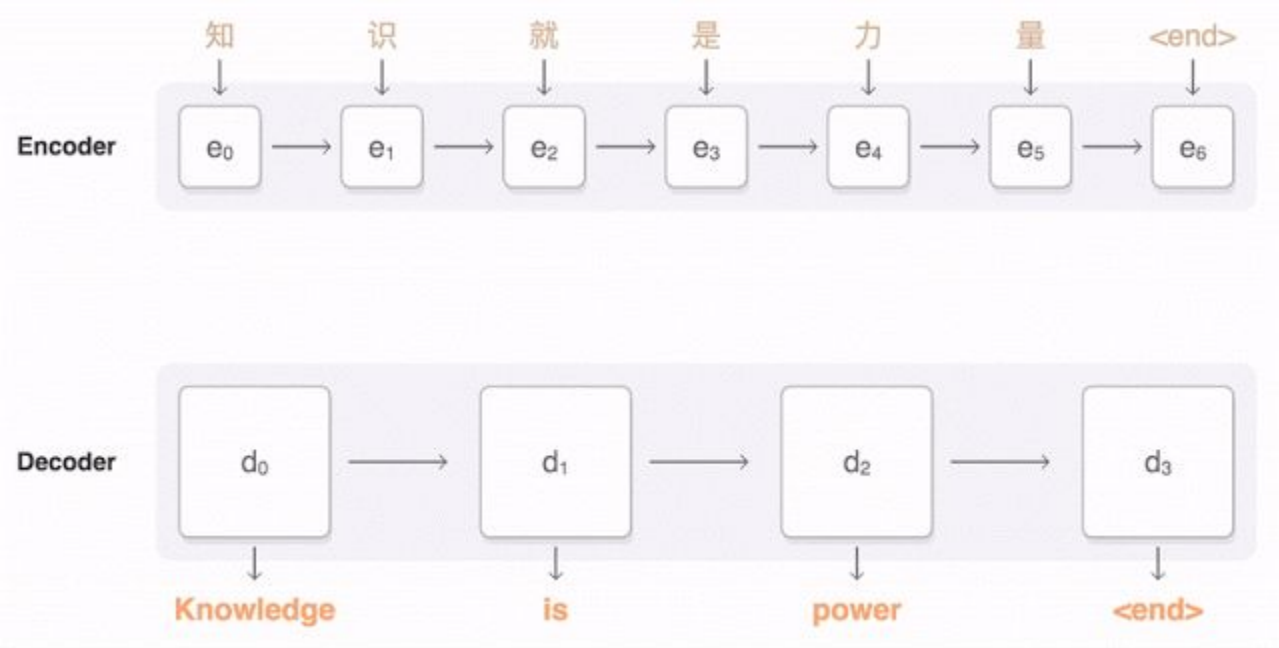
Distribution Alignment Problem

- Image to Image

(Isola et al, 2017)

Distribution Alignment Problem

- Text to text



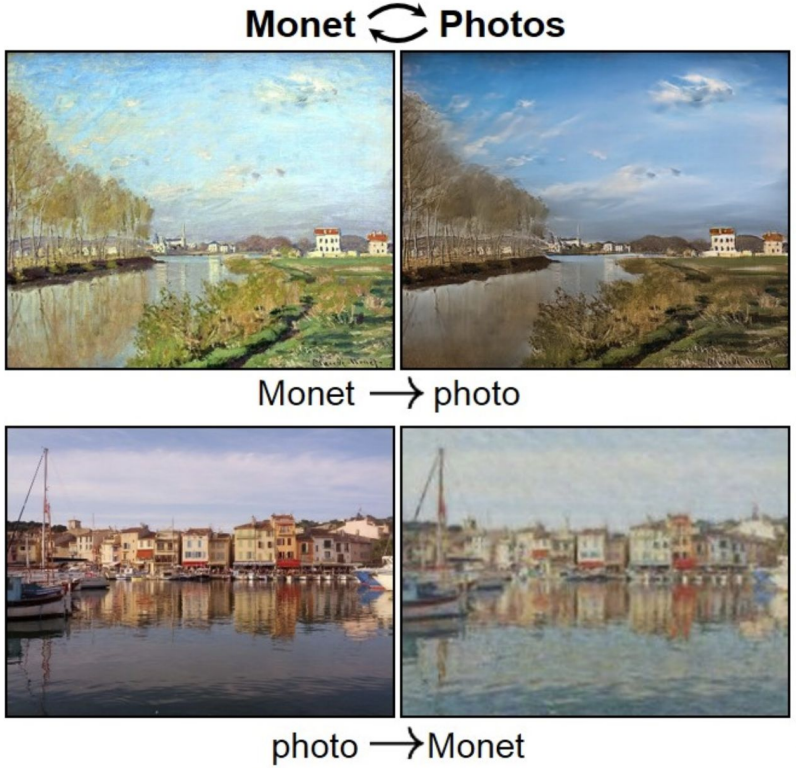
https://ai.googleblog.com/2016/09/a-neural-network-for-machine.html

Supervised Distribution Alignment

- Image to Image: pix2pix (Isola et al, 2017)
- Text to text: machine translation
- Image to text: captioning
- Text to Image, voice to text, text to voice….
- It's simply fitting conditional distribution p(a|b)
  - We have access to (a, b) pairs
  - What if (a, b) pairs are expensive to obtain or just don't exist?

Unsupervised Distribution Alignment

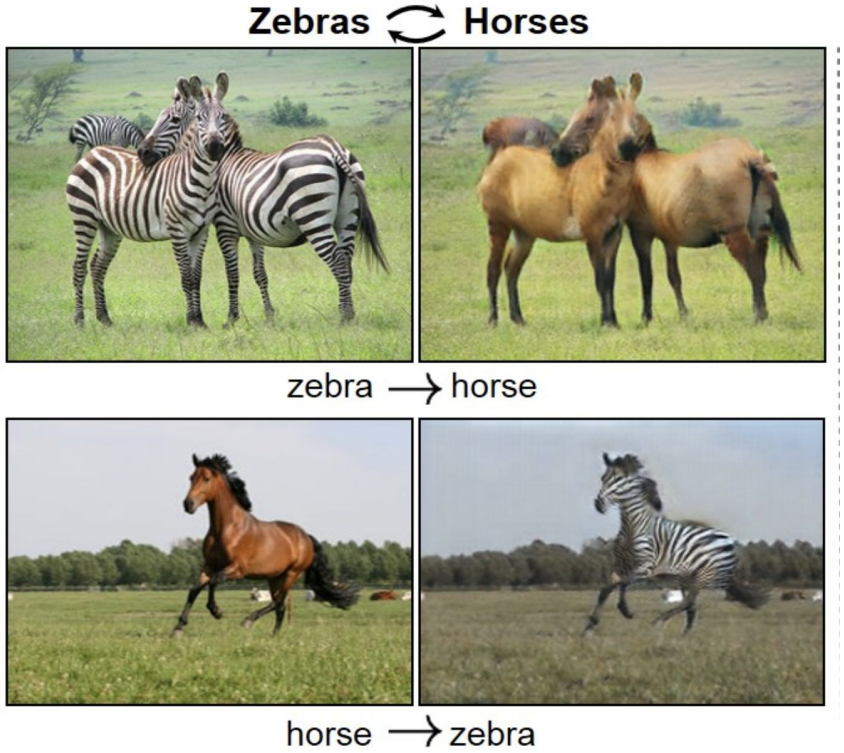
- Image to Image



(Zhu et al, 2017)

Unsupervised Distribution Alignment

- Image to Image



(Zhu et al, 2017)

Unsupervised Distribution Alignment

- A lot of applications:
  - translation to/from small languages that are not economical to label
  - augment labeled examples
  - style transfer etc
- Is this even a feasible problem?
  - We have access to samples from p(a) and p(b)
  - Without any access to samples from p(a, b) we need to estimate p(a|b) and p(b|a)

Marginal Matching

- We will try to learn the relationship between A and B:
  - Approximate p(a|b) by q_θ(a|b) and p(b|a) by q_θ(b|a)
- The marginals induced by approximate mapping q should match original margins:

$$q(b) = \mathbb{E}_{a \sim p(a)} \left[q(b|a) \right] \approx \mathbb{E}_{a \sim p(a)} \left[p(b|a) \right] = p(b)$$

$$q(a) = \mathbb{E}_{b \sim p(b)} \left[q(a|b) \right] \approx \mathbb{E}_{b \sim p(b)} \left[p(a|b) \right] = p(a)$$

- In literature, q(b|a) is oftentimes just a deterministic mapping, which we call G_{AB} : A -> B

Marginal Matching

- [hand-draw 1d example]

Marginal Matching

- [1d example, ambiguity]

Cycle Consistency

- Many names in the literature: Cycle Consistency, Dual Learning, Back translation, …
- Core idea:

$$\mathbb{E}_{b\sim q(b|a)}\left[q(a'|b)\right] \approx \mathbb{E}_{b\sim p(b|a)}\left[p(a'|b)\right] \forall a$$

- In the case of deterministic mapping:

$$G_{BA}(G_{AB}(a)) = a$$
$$G_{AB}(G_{BA}(b)) = b$$

Marginal Matching

- [1d example, with reduced ambiguity]

(partial) Unsupervised Alignment Principles

- We have come up with two invariances that are true for all alignment problems and we can use them as learning signals
  - Marginal Matching
  - Cycle Consistency
- These are obviously not enough in general.
  - In practice, researchers inject additional inductive biases into learning systems by selecting architectures, loss functions, and problems.

CycleGAN

- Marginal matching w/ GAN

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))], \quad (1)$$

- Cycle consistency w/ deterministic mapping & L1 loss

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].$$

(Zhu et al, 2017)

CycleGAN

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [32]	0.45	0.11	0.08
BiGAN/ALI [9, 7]	0.41	0.13	0.07
SimGAN [46]	0.47	0.11	0.07
Feature loss + GAN	0.50	0.10	0.06
CycleGAN (ours)	**0.58**	**0.22**	**0.16**
pix2pix [22]	0.85	0.40	0.32

Table 3: Classification performance of photo→labels for different methods on cityscapes.

(Zhu et al, 2017)

CycleGAN

Loss	Per-pixel acc.	Per-class acc.	Class IOU
Cycle alone	0.10	0.05	0.02
GAN alone	0.53	0.11	0.07
GAN + forward cycle	0.49	0.11	0.07
GAN + backward cycle	0.01	0.06	0.01
CycleGAN (ours)	**0.58**	**0.22**	**0.16**

Table 5: Ablation study: classification performance of photo→labels for different losses, evaluated on Cityscapes.

(Zhu et al, 2017)

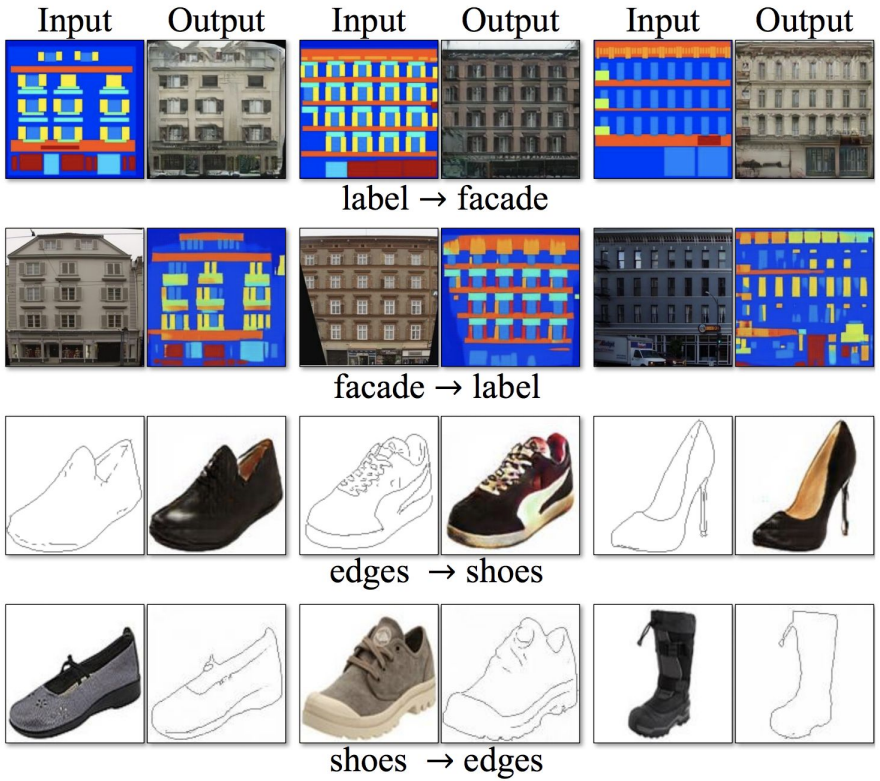
CycleGAN

Loss	Per-pixel acc.	Per-class acc.	Class IOU
Cycle alone	0.22	0.07	0.02
GAN alone	0.51	0.11	0.08
GAN + forward cycle	**0.55**	**0.18**	**0.12**
GAN + backward cycle	0.39	0.14	0.06
CycleGAN (ours)	0.52	0.17	0.11

Table 4: Ablation study: FCN-scores for different variants of our method, evaluated on Cityscapes labels→photo.

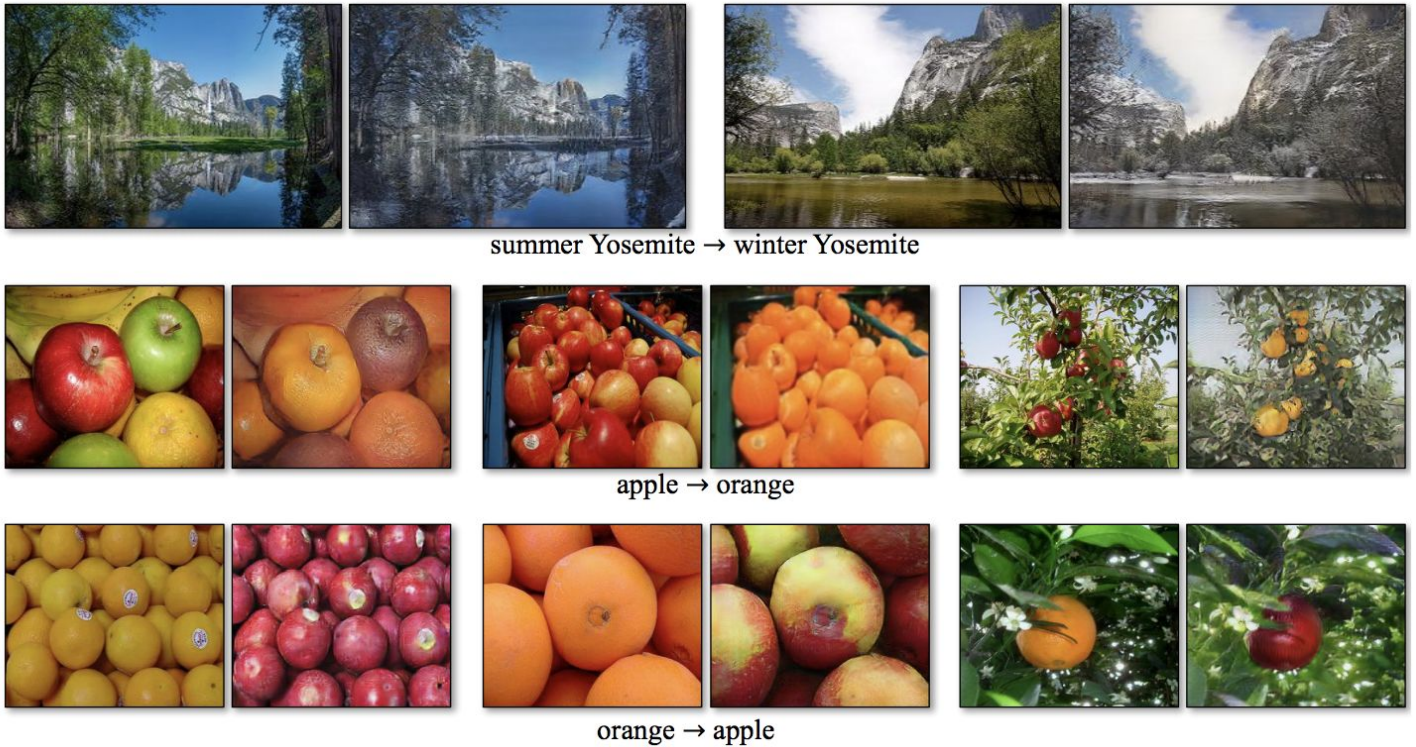
(Zhu et al, 2017)

CycleGAN



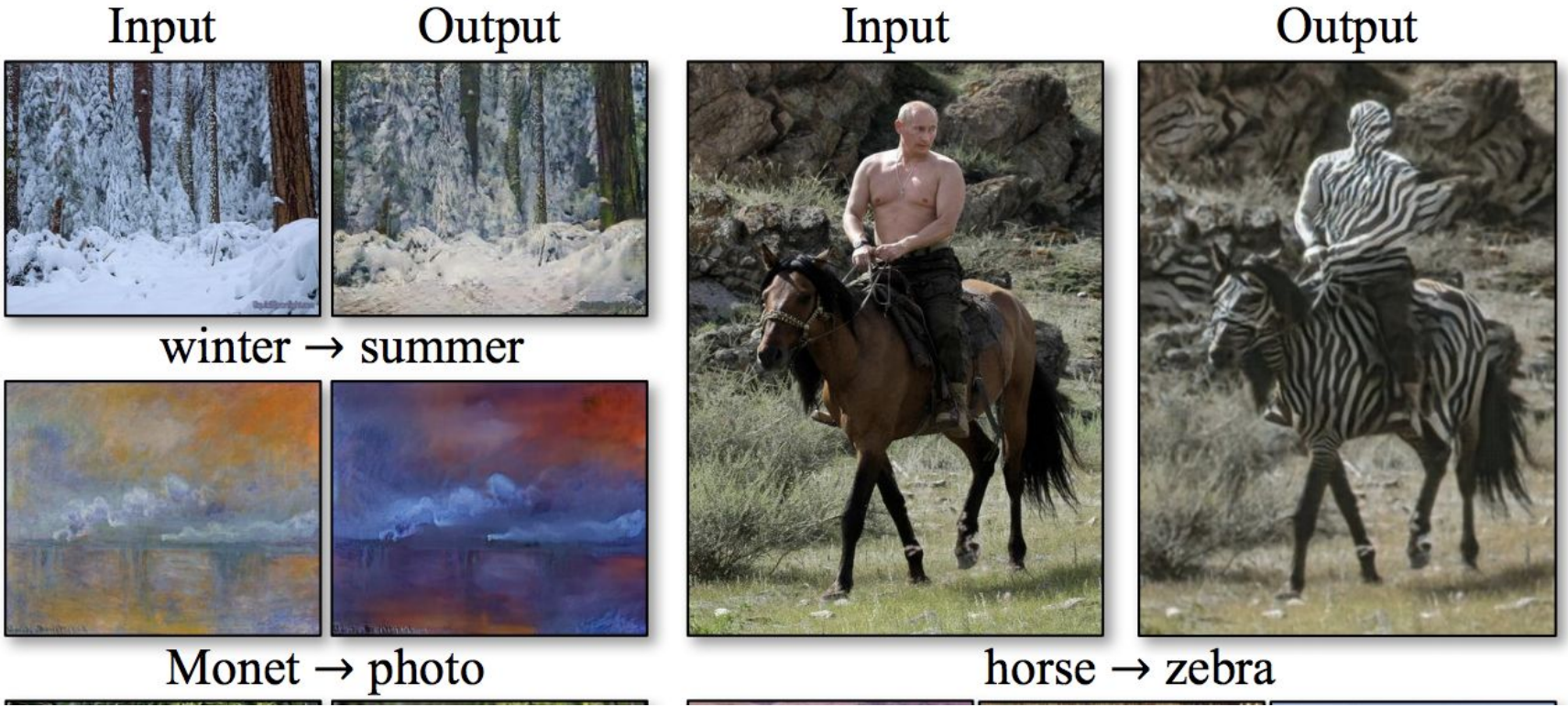
(Zhu et al, 2017)

CycleGAN



(Zhu et al, 2017)

CycleGAN failure cases



(Zhu et al, 2017)

Stochastic Mapping

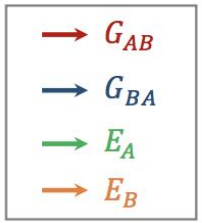
- Sometimes deterministic (one-to-one) mappings are too restricted, e.g. semantic mask <> image
- One straightforward way to extend CycleGAN is to make the mapping take in an additional noise source (DualGAN)

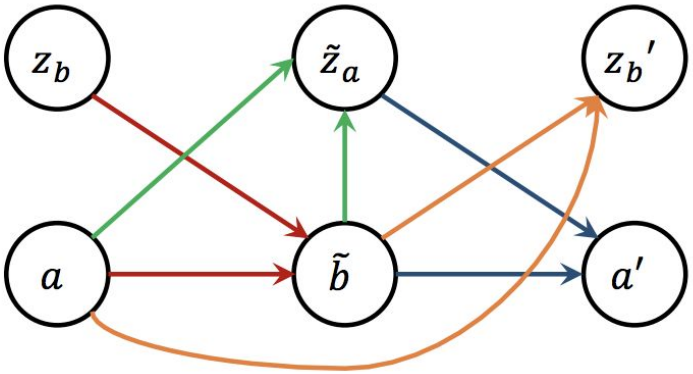
$$G_{AB}(a, z) : A \times Z \rightarrow B \quad \text{instead of } G_{AB}(a) : A \rightarrow B$$

- However, we also need to change cycle-consistency l1 loss since otherwise z will simply be ignored
  - $G_{BA}(G_{AB}(a, z), z') = a$, which means the choice of z, z' is irrelevant

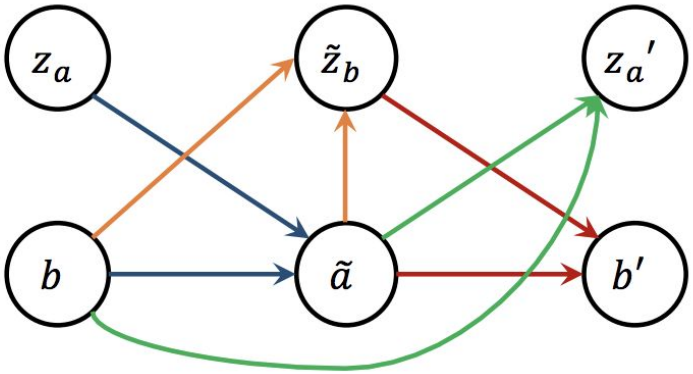
(Almahairi et al, 2017)

Augmented CycleGAN





Cycle starting from $A \times Z_b$



Cycle starting from $B \times Z_a$

(Almahairi et al, 2017)

Augmented CycleGAN

$$\mathcal{L}^{A}_{\text{CYC}}(G_{AB}, G_{BA}, E_A) = \mathop{\mathbb{E}}_{\substack{a \sim p_d(a) \\ z_b \sim p(z_b)}} \left\| a' - a \right\|_1,$$

$$\tilde{b} = G_{AB}(a, z_b), \;\; \tilde{z}_a = E_A(a, \tilde{b}), \;\; a' = G_{BA}(\tilde{b}, \tilde{z}_a). \quad (9)$$

(Almahairi et al, 2017)

CycleGAN can "cheat"

- It "can" generate diverse mappings

(a) AugCGAN (b) StochCGAN

Figure 5: Given an edge from the data distribution (leftmost column), we generate shoes by sampling five $z_b \sim p(z_b)$. Models generate diverse shoes when edges are from the data distribution.

(Almahairi et al, 2017)

CycleGAN can "cheat"

- And be cycle-consistent at the same time

(c) AugCGAN (d) StochCGAN

Figure 6: Cycles from both models starting from a real edge and a real shoe (left and right respectively in each subfigure). The ability for StochCGAN to reconstruct shoes is surprising and is due to the "steganography" effect (see text).

(Almahairi et al, 2017)

Augmented CycleGAN

(a) AugCGAN

(b) StochCGAN

(Almahairi et al, 2017)

Augmented CycleGAN

(a) AugCGAN

(b) StochCGAN

(Almahairi et al, 2017)

Can we do better?

- Can we do better than only relying on 1) Marginal matching and 2) Cycle consistency?
  - Not clear what other invariances we can rely on (good open problem)
- Core problem is that aligning p(a) and p(b) without knowing what's inside a & b is too difficult.
  - One direction: a & b are usually high-dimensional; there is additional structure in them
  - (current image-image alignment works that use ConvNet or patch-based discriminator are already implicitly using this principle)

An NLP Example

- Let's say A = all english sentences, B = all french sentences
  - Two semantically unrelated sentences a, b might have the same frequency p(a) = p(b) and cycle-consistency won't rule that out either
  - Nevertheless, we know each sentence is made up of words and it's unlikely the words in those two unrelated sentences have the same statistics.
- Here we start to make additional assumption, sub-components (e.g. words) of a large random variable (e.g. sentences) can have their own alignment.
  - And we can make use of co-occurrence statistics of the sub-components. word2vec!

recap: word2vec - Skip Gram

Skip-gram model

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_O|w_I)=\frac{\exp\left({v'_{w_O}}^{\top}v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^{\top}v_{w_I}\right)}$$

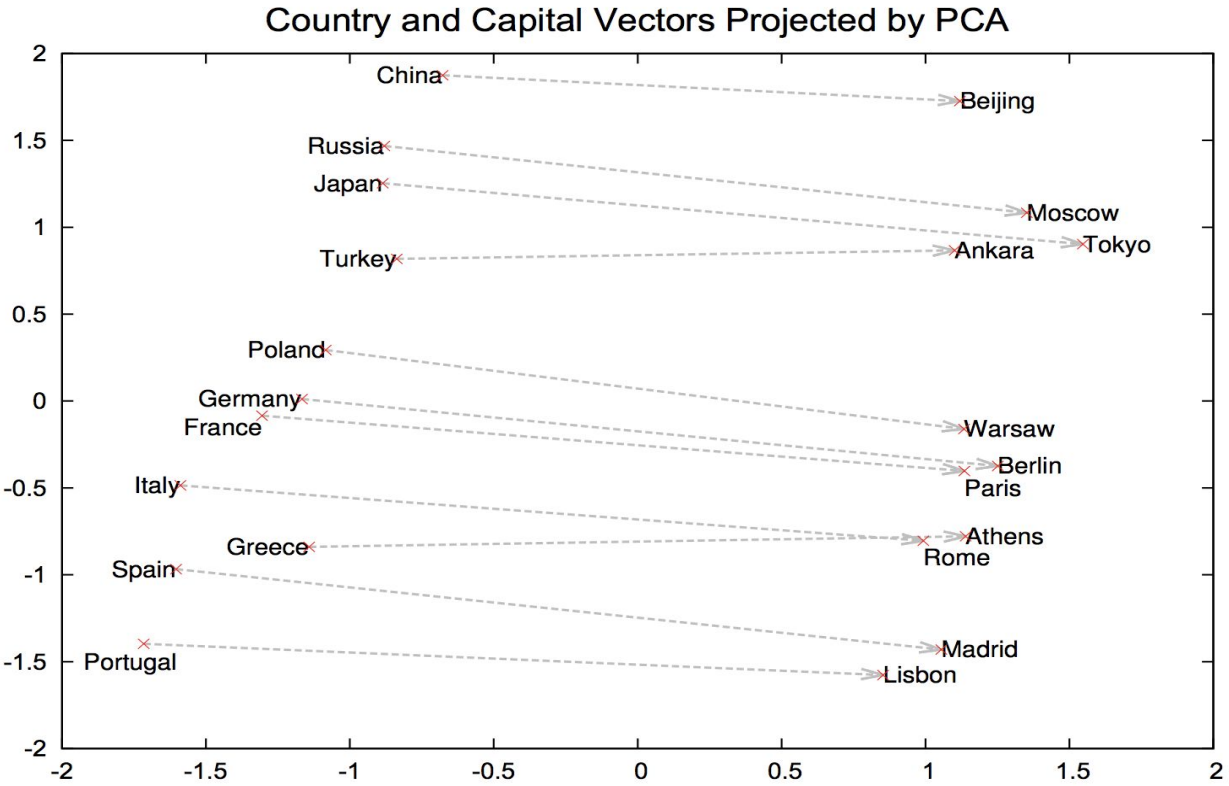
Don't have to have the denominator over all words in the vocabulary

- Can use negative sampling

$$\log\sigma({v'_{w_O}}^{\top}v_{w_I})+\sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\sigma(-{v'_{w_i}}^{\top}v_{w_I})\right]$$

$$P(w_i)=1-\sqrt{\frac{t}{f(w_i)}}$$

recap: word2vec



recap: word2vec

	NEG-15 with 10^{-5} subsampling	HS with 10^{-5} subsampling
Vasco de Gama	Lingsugur	Italian explorer
Lake Baikal	Great Rift Valley	Aral Sea
Alan Bean	Rebbeca Naomi	moonwalker
Ionian Sea	Ruegen	Ionian Islands
chess master	chess grandmaster	Garry Kasparov

Table 4: Examples of the closest entities to the given short phrases, using two different models.

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

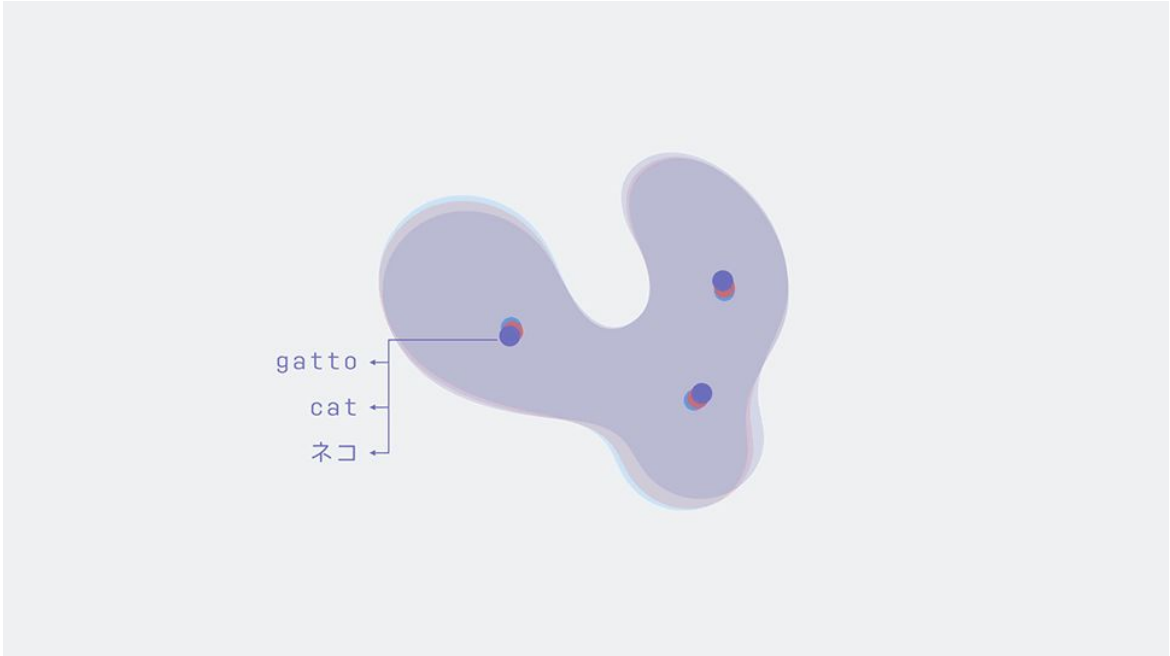
word2vec alignment

- If similar vector calculus holds in all languages, can we align word embeddings in different languages just by uncovering some affine transformation?

(Facebook blog)

word2vec alignment

- Magically it's only a rotation away (Mikolov et al. 2013, Xing et al 2015)



(Facebook blog)

Unsupervised Word Alignment

- (Conneau et al. 2018) proposed the following alignment algorithm
  a. Approximate marginal matching by adversarial training
  b. Refined rotation by solving for exact alignment from a) top pairs

Unsupervised Word Alignment

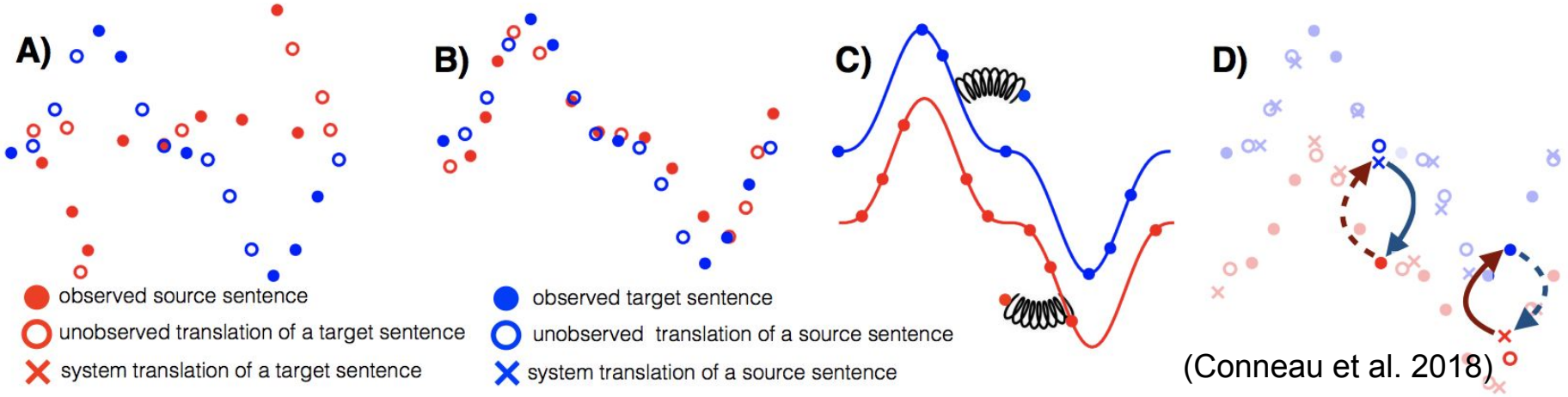
	en-es	es-en	en-fr	fr-en	en-de	de-en	en-ru	ru-en	en-zh	zh-en	en-eo	eo-en
Methods with cross-lingual supervision and fastText embeddings												
Procrustes - NN	77.4	77.3	74.9	76.1	68.4	67.7	47.0	58.2	40.6	30.2	22.1	20.4
Procrustes - ISF	81.1	82.6	81.1	81.3	71.1	71.5	49.5	63.8	35.7	**37.5**	29.0	27.9
Procrustes - CSLS	81.4	82.9	81.1	**82.4**	73.5	**72.4**	**51.7**	**63.7**	**42.7**	36.7	**29.3**	25.3
Methods without cross-lingual supervision and fastText embeddings												
Adv - NN	69.8	71.3	70.4	61.9	63.1	59.6	29.1	41.5	18.5	22.3	13.5	12.1
Adv - CSLS	75.7	79.7	77.8	71.2	70.1	66.4	37.2	48.1	23.4	28.3	18.6	16.6
Adv - Refine - NN	79.1	78.1	78.1	78.2	71.3	69.6	37.3	54.3	30.9	21.9	20.7	20.6
Adv - Refine - CSLS	**81.7**	**83.3**	**82.3**	82.1	**74.0**	72.2	44.0	59.1	32.5	31.4	28.2	**25.6**

Table 1: Word translation retrieval P@1 for our released vocabularies in various language pairs. We consider 1,500 source test queries, and 200k target words for each language pair. We use fastText embeddings trained on Wikipedia. NN: nearest neighbors. ISF: inverted softmax. ('en' is English, 'fr' is French, 'de' is German, 'ru' is Russian, 'zh' is classical Chinese and 'eo' is Esperanto)

(Conneau et al. 2018)

Unsupervised Machine Translation

- (Lample et al. 2018) leverages all 3 core principles:
  - word-level alignment (= sub-component level statistics)
  - monolingual language models (= marginal matching)
  - back translation (= cycle consistency)



(Conneau et al. 2018)

Unsupervised Machine Translation

	en → fr	fr → en	en → de	de → en	en → ro	ro → en	en → ru	ru → en
Unsupervised PBSMT								
Unsupervised phrase table	-	17.50	-	15.63	-	14.10	-	8.08
Back-translation - Iter. 1	24.79	26.16	15.92	22.43	18.21	21.49	11.04	15.16
Back-translation - Iter. 2	27.32	26.80	17.65	22.85	20.61	22.52	12.87	16.42
Back-translation - Iter. 3	27.77	26.93	17.94	22.87	21.18	22.99	13.13	16.52
Back-translation - Iter. 4	27.84	27.20	17.77	22.68	21.33	23.01	13.37	**16.62**
Back-translation - Iter. 5	**28.11**	27.16	-	-	-	-	-	-
Unsupervised NMT								
LSTM	24.48	23.74	14.71	19.60	-	-	-	-
Transformer	25.14	24.18	17.16	21.00	21.18	19.44	7.98	9.09
Phrase-based + Neural network								
NMT + PBSMT	27.12	26.29	17.52	22.06	21.95	23.73	10.14	12.62
PBSMT + NMT	27.60	**27.68**	**20.23**	**25.19**	**25.13**	**23.90**	**13.76**	**16.62**

(Conneau et al. 2018)

Unsupervised Machine Translation

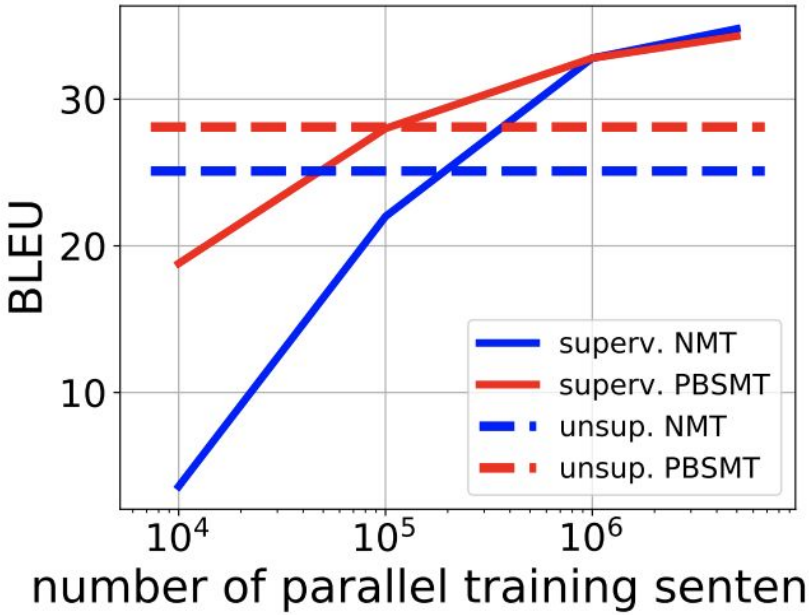


Figure 2: Comparison between supervised and unsupervised approaches on WMT'14 En-Fr, as we vary the number of parallel sentences for the supervised methods.

(Conneau et al. 2018)