Domain Adaptation for Machine Translation

Kevin Duh

Johns Hopkins University

May 2018

Domain Adaptation: Machine Learning Perspective

- Training data:
  - (x_1,y_1), (x_2,y_2), (x_3,y_3), …, i.i.d. samples from distribution ***D***
  - Build model $p(y|x)$
- Test data x_t:
  - If x_t is not from ***D***, $p(y|x=x_t)$ is operating at an input space it wasn't built for.

- Youtube example of NMT operating on sentences it doesn't expect:
- https://www.youtube.com/watch?v=3-rfBsWmo0M

Domain Adaptation: Terminology

- Supervised adaptation
  - Given large Out-of-Domain (x,y), small In-Domain (x,y)
  - Test on in-domain data
- Unsupervised adaptation
  - Given only large Out-of-Domain (x,y)
  - Test on in-domain data
- Semi-supervised adaptation
  - Given large Out-of-Domain (x,y) and In-Domain (x)
  - Test on in-domain data

Switch to Philipp's slides

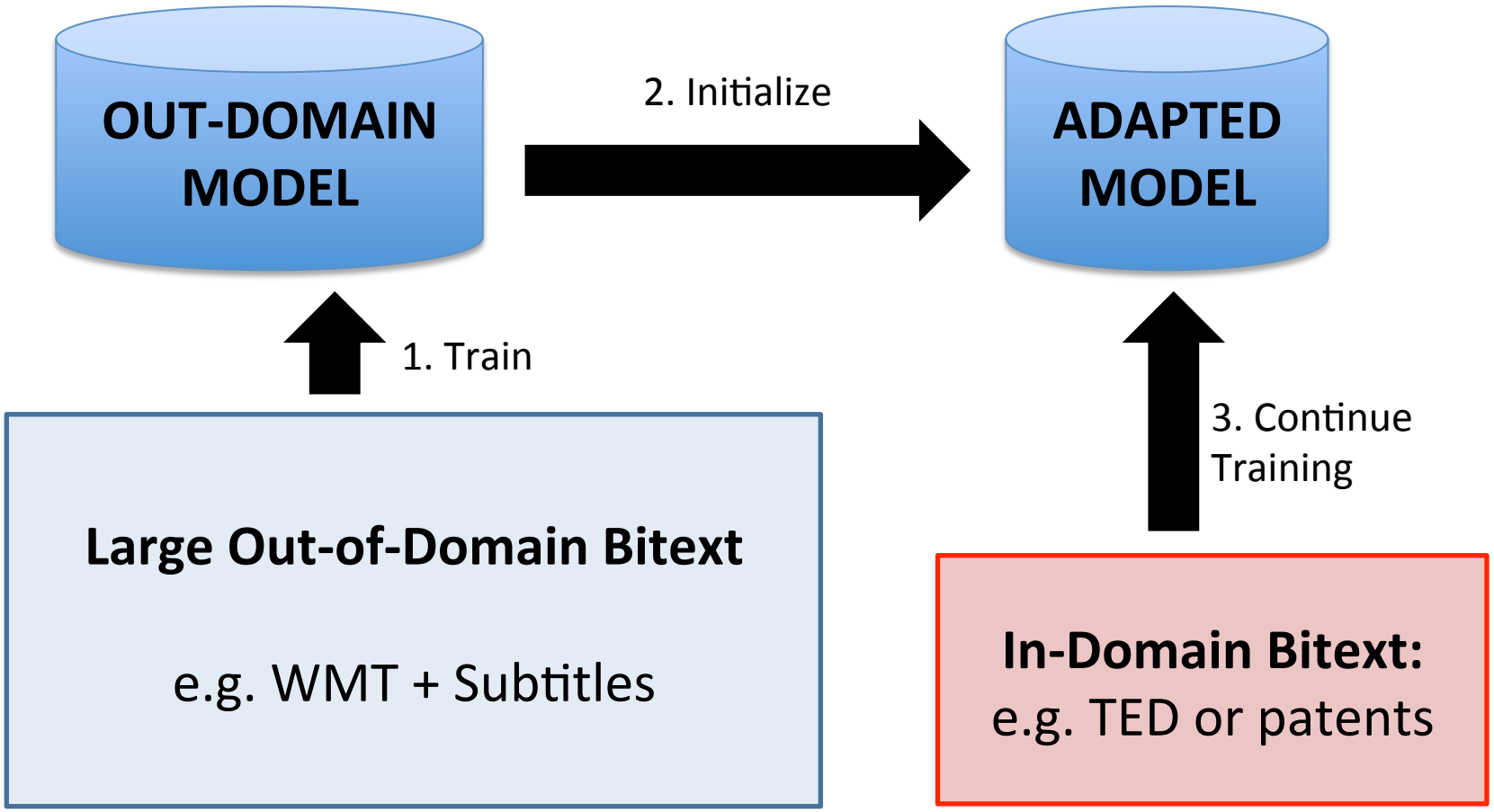
- http://mt-class.org/jhu/slides/lecture-domain-adaptation.pdf

General classes of algorithms for Adaptation for NMT

- Fine Tuning / Continued Training
  - Catastrophic forgetting issues, addressed via e.g. learning rate adjustment or regularization
  - Data sparsity issues, addressed via e.g. partial model updates
- Instance Weighting
- Add a tag to input indicating domain
  - Similar to multisource translation

Quick survey of example algorithms..

NMT Fine Tuning / Continued Training



BLEU Results: OUT -> {TED, Patent}

Test	Training data for NMT model	Ar-En	De-En	Fa-En	Ko-En	Ru-En	Zh-En
TED Talks	Out Domain	25.7	30.0	17.2	7.5	21.7	13.6
	In Domain (TED)	21.7	25.4	15.6	9.9	16.0	11.2
	Continue Training	30.7	34.3	22.0	12.6	25.5	16.1
Patent	Out Domain	n/a	25.2	n/a	1.5	18.9	11.6
	In-Domain (Patent)	n/a	54.3	n/a	19.8	6.8	28.0
	Continue Training	n/a	55.6	n/a	22.6	28.1	32.9

An Empirical Comparison of Simple Domain Adaptation Methods for Neural Machine Translation

Chenhui Chu[1] , **Raj Dabre**[2], and **Sadao Kurohashi**[2]
[1]Japan Science and Technology Agency
[2]Graduate School of Informatics, Kyoto University

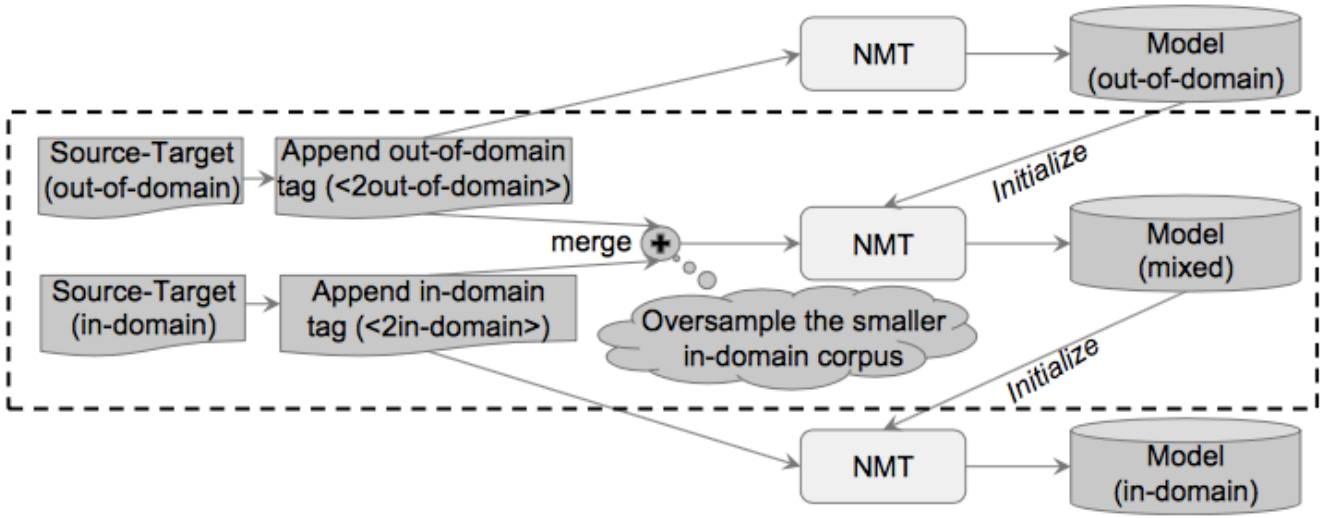


Figure 2: Mixed fine tuning with domain tags for domain adaptation (The section in the dotted rectangle denotes the multi domain method).

Regularized Training Objective for Continued Training for Domain Adaption in Neural Machine Translation

Huda Khayrallah Brian Thompson Kevin Duh Philipp Koehn
Department of Computer Science
Johns Hopkins University

$$\mathcal{L}_{\text{NLL}}(\theta) = -\sum_{v \in \mathcal{V}} \big(\mathbb{1}\{y_i = v\} \times \log\ p(y_i = v \,|\, x; \theta; y_{j<i}) \big) \tag{1}$$

$$\mathcal{L}_{\text{reg}}(\theta) = -\sum_{v \in \mathcal{V}} \big(p_{aux}(y_i = v \,|\, x; \theta_{aux}; y_{j<i}) \times \log p(y_i = v \,|\, x; \theta; y_{j<i}) \big) \tag{2}$$

$$\mathcal{L}(\theta) = (1 - \alpha)\ \mathcal{L}_{\text{NLL}}(\theta) + \ \alpha\ \mathcal{L}_{\text{reg}}(\theta)$$

Instance Weighting

- Weight in-domain and out-domain samples differently

Instance Weighting for Neural Machine Translation Domain Adaptation

Rui Wang[1], **Masao Utiyama**[1], **Lemao Liu**[2], **Kehai Chen**[1,3] **and Eiichiro Sumita**[1]
[1]National Institute of Information and Communications Technology (NICT)
[2]Tencent AI Lab
[3]Harbin Institute of Technology

$$J_{sw} = \sum_{\langle \mathbf{x}_i, \mathbf{y}_i \rangle \in \mathcal{D}} \lambda_i \log p(\mathbf{y}_i|\mathbf{x}_i).$$

$$J_{dw} = \lambda_{in} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_{in}} logp(\mathbf{y}|\mathbf{x}) + \sum_{(\mathbf{x}',\mathbf{y}') \in \mathcal{D}_{out}} logp(\mathbf{y}'|\mathbf{x}').$$

Cost Weighting for Neural Machine Translation Domain Adaptation

Boxing Chen, Colin Cherry, George Foster, and Samuel Larkin
National Research Council Canada
Ottawa, ON, Canada

$$\theta^{\star} = \arg\max_{\theta} \sum_{(x,y)\in D} (1 + p_d(x)) \log p(y|x;\theta)$$

$$p_d(x) = \sigma\left(\tanh\left(W^d r_x + b^d\right)^{\top} w^d\right)$$

$$\text{where } \sigma(x) = \frac{1}{1+\exp(-x)}$$

Summary

- Domain adaptation: training data and test data are from different distributions
- Supervised adaptation: give large out-of-domain (x,y) and small in-domain (x,y)
  - Generic Approaches: Concatenate data, Mixture model, Subsampling
  - SMT-specific: Phrase-table back-off
  - NMT-specific: continue training

Open Problems

- How to quantify domain characteristics?
- Model selection in realistic settings?
- Other approaches for SMT, NMT adaptation
- Adapt to documents, speakers, style

- If we solve Domain Adaptation, we'll be much closer to enabling ubiquitous MT!