CS769 Fall 2023

Multi-task Learning in NLP

Jiang, Yuye

PhD student in Computer Sciences

Slides adapted from: https://junjiehu.github.io/cs769-fall23/assets/pdf/anlp-15-multitask.pdf

Goals for Today

Two scenarios of Multi-task Learning:

- Multi-domain Learning
- Multi-lingual Learning

Three Categories of Methods

- (Model) Parameter-sharing / Invariant Feature Learning
- (Learning) Task Re-weighting
- (Data) Data augmentation

Different "tasks"

Two scenarios of Multi-task Learning:

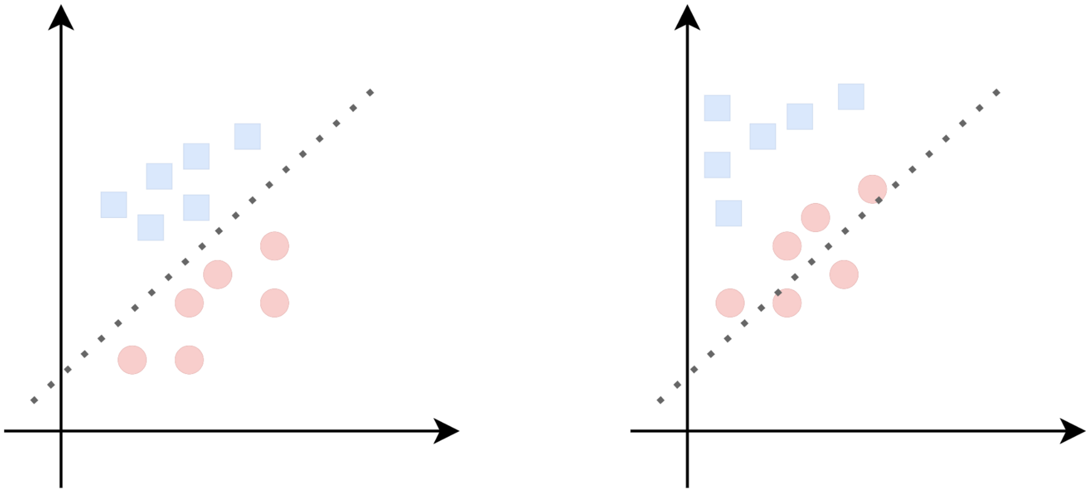
- Task in different domain: Multi-domain Learning
- For NLP: Task in different language: Multi-lingual Learning

Domain in Machine Learning

Mathematically:

In two different domains, joint distribution over inputs (X) and outputs (Y) differs

$$P_{d1}(X, Y) \neq P_{d2}(X, Y)$$



(Stewart 2019)

Domain in NLP

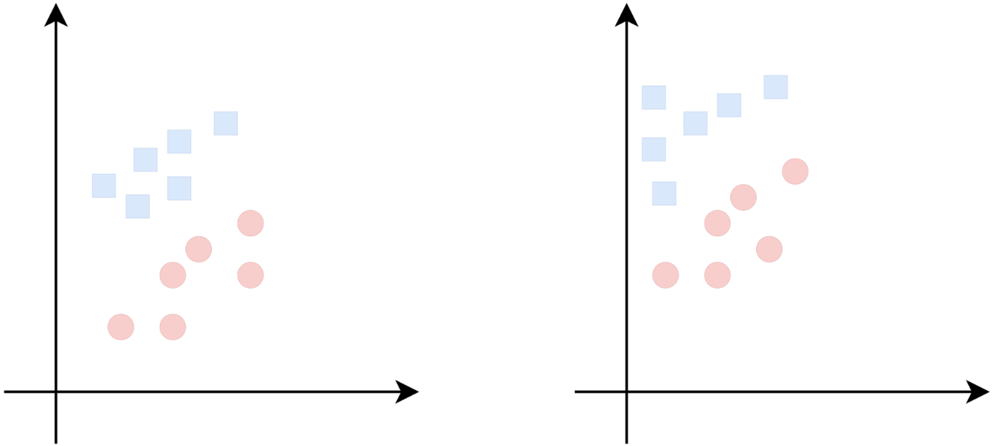
Practically:

- **Content**: what is being discussed
  - Text from medical domain versus Text from financial domain
  - Terminology, term frequency
- **Style**: the way in which it is being discussed
  - Text from social media versus Text from newspaper
- **Labeling Standards**: the way that the same data is labeled
  - Same sentence, labeled by different population

Domain Shift

- **Covariate Shift:** (Discriminative models) Input changes, but not the labeling

$$P_{d1}(X) \neq P_{d2}(X) \qquad P_{d1}(Y|X) = P_{d2}(Y|X)$$

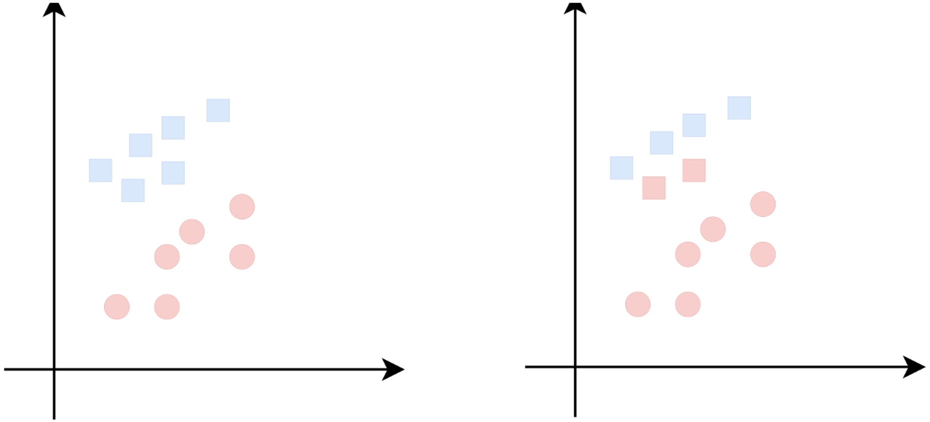


In NLP: Speech recognition for American English speakers and British English speakers

Domain Shift

- **Concept Shift**: (Discriminative models) Input distribution same, conditional distribution changes

$$P_{d1}(X) = P_{d2}(X) \qquad P_{d1}(Y|X) \neq P_{d2}(Y|X)$$



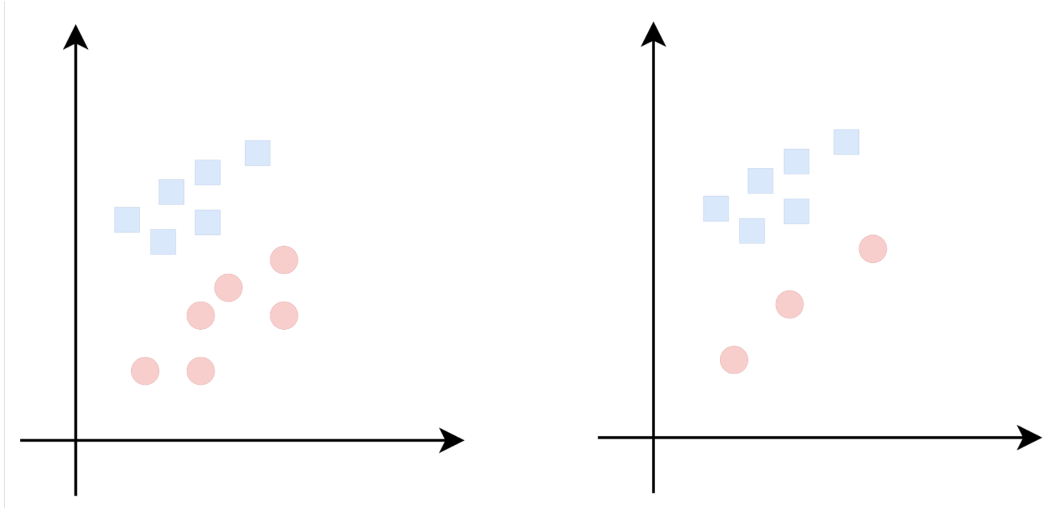
In NLP: Words with different meaning in different places

窝心 wōxīn in North China = feel happy, in South China = feel unhappy

Domain Shift

- **Label Shift**: (Generative models) The label distribution P(Y) changes

$$P_{d1}(Y) \neq P_{d2}(Y)$$



In NLP: spam filter trained on 50/50, applied on 95/5 in real world

Out of Distribution / Domain (OOD)

- **Generalization**
  - **Domain adaptation**: train on some domain, adapt to a target domain at testing and perform well on that target domain
  - **Domain robustness**: train on some domain, perform well on all domains including especially minority domains

- **Out of Distribution Detection**
  - On classification task: detect whether a test example is an OOD example or not

Coling. 2020. Neural Unsupervised Domain Adaptation in NLP—A Survey

Domain adaptation in NLP

Why multitasking: When one of your domain has fewer data, adaptation improves performance

Adaptation from: More data → Less data

- Plain text → labeled text
  (Language Modeling) → (Sentiment Analysis)
- General domain → specific domain
  (Web text) → (Medical text)
- High-resourced language → low-resourced language
  (English) → (Telugu)

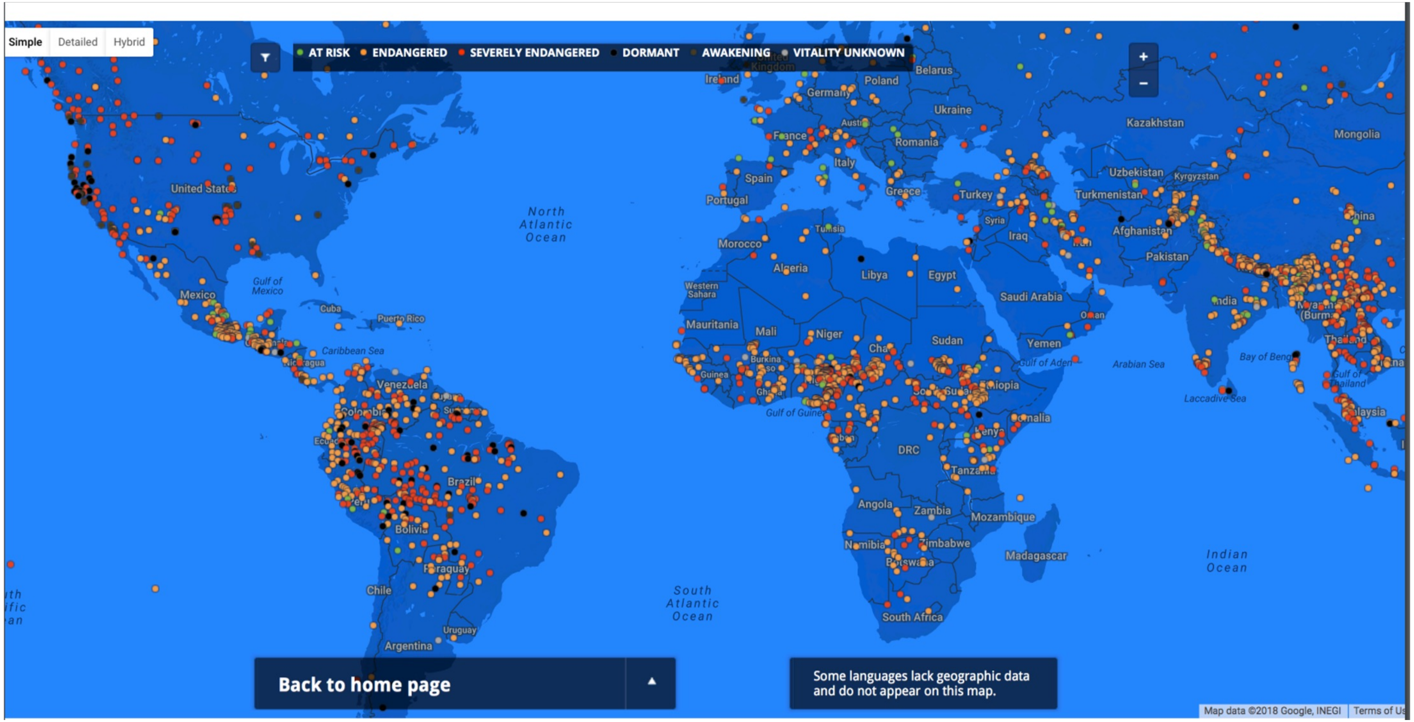
Types:

- **Supervised adaptation**: train with some target domain data
- **Unsupervised adaptation**: train without target domain data

Domain robustness in NLP

- Train a generic model, may be applied to different domains
- Robustness in minority domains
- **Zero-shot** robustness to domains not in training data

Multilingual Learning



https://endangeredlanguages.com/

Similarities across languages

- Similar word roots

Cognates (joint origin)

English:	night
French:	nuit
Russian:	noch
Bengali:	nishi

Loan Words (borrowed from another)

Arabic:	qahwa
Turkish:	kahveh
English:	coffee
Japanese:	kohi
Chinese:	kafei

- Similar grammar

Domain(Language) adaptation in NLP

- **Domain adaptation**: train on high-resource languages, improve accuracy by transferring to low-resource languages
- **Domain robustness**: Train one model for all languages, instead of one model for each; workwell on unseen / low-resource languages / dialect / nonstandard text / typos etc

Challenges: similarities & differences in lexicon, morphology, syntax, semantics, culture

Multi-tasking methods

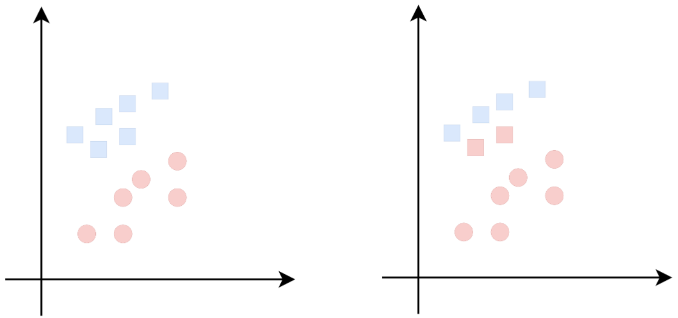
- **(Model) Parameter-sharing / Invariant Feature Learning**
- (Learning) Task Re-weighting
- (Data) Data augmentation

Parameter Sharing

- All parameters
  - one single model for all domains
  - e.g. Same GPT4 model for many tasks
- Some parameters
  - shared encoder, separate decoder (many unshared)
  - shared encoder, different task head (very little unshared)
    - e.g. BERT with different head

All parameters

- Ignore domain differences, just train a single model
- Multilingually
  - Multilingual MT to English (Neubig and Hu 2018)
  - Multilingual pre-trained LMs (Devlin et al. 2019, Wu and Dredze 2019)
- Cannot achieve idea accuracy under **concept shift**



Parameter decoupling: Domain Tag

- Append a domain tag to input (Chu et al. 2017)

<news> news text

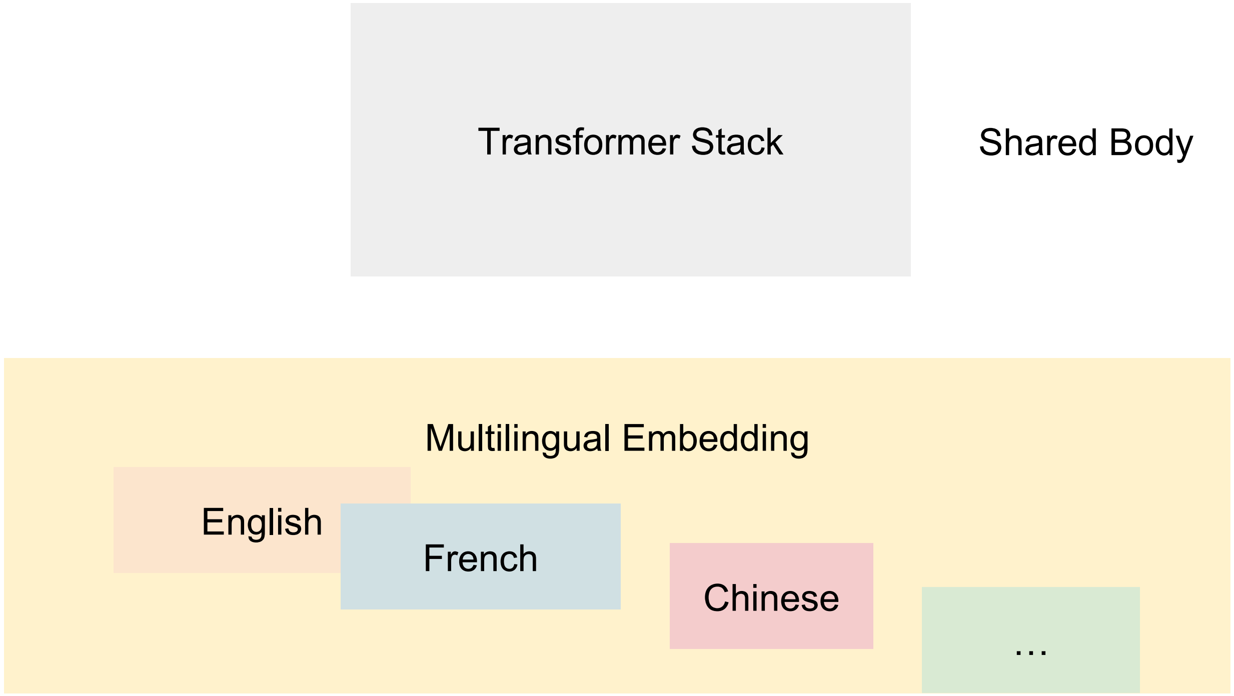
<med> medical text

- Append a language tag of the target output to same input (Johnson et al. 2017)

<fr> this is an example → ceci est un exemple

<ja> this is an example → これは例です

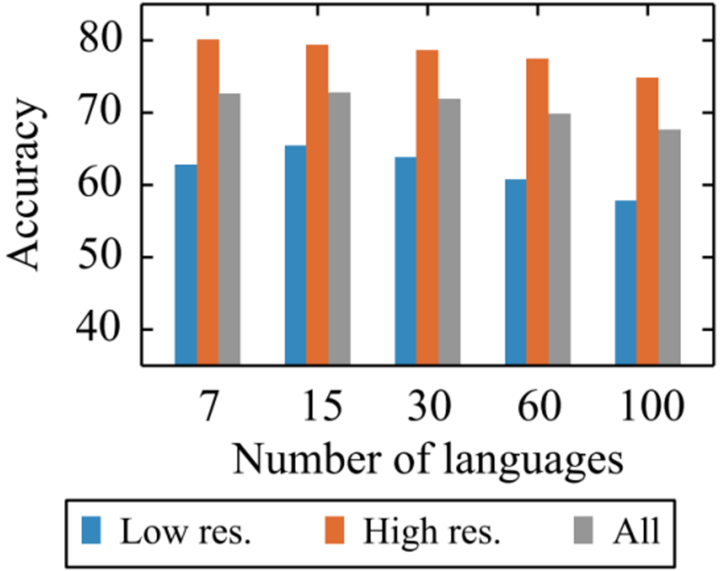
Parameter decoupling: Multilingual Vocabulary



Curse of Multilinguality

- In a fixed sized model, the per-language capacity decreases as we increase the number of languages

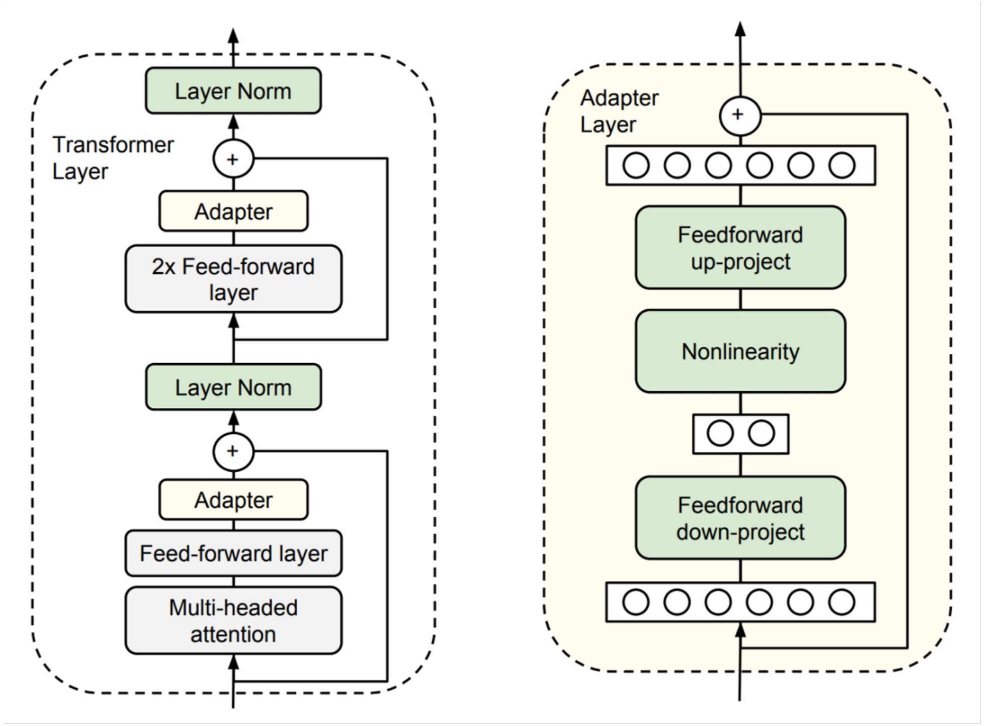
- Increasing the number of languages → decrease in the quality of all language accuracy (Conneau et al. 2019)



Parameter decoupling: Adapters

- Add a small layer per task to the already trained model

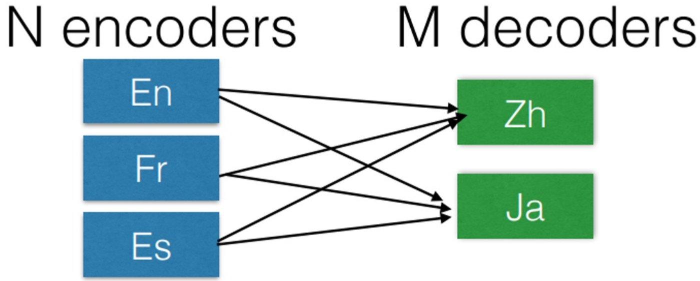
- Freeze the parameters of already trained part, only update the adapters on different domain data



(Houlsby et al. 2019)

Parameter decoupling: Aggressive Parameter Decoupling

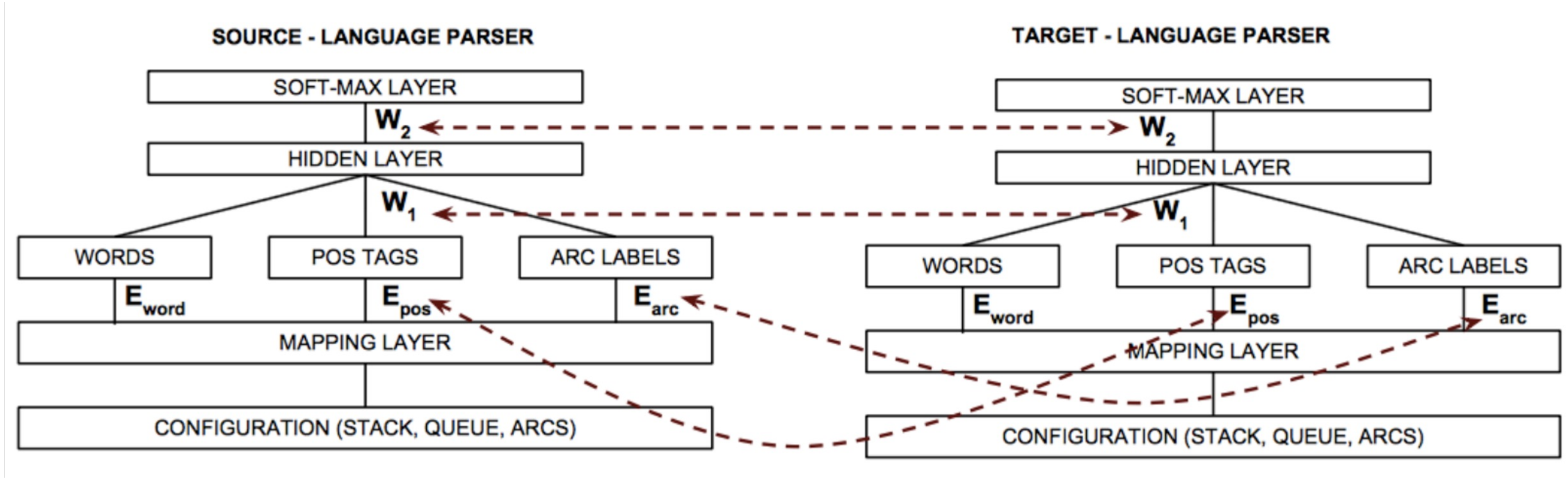
- Multilingual Machine Translation: one encoder or decoder per language (Firat et al. 2016)



Problem: Explosion in number of parameters

Soft Parameter Tying for Multi-task Learning

- Share parameters loosely between various tasks
- Parameters are regularized to be closer, but not tied in a hard fashion (e.g. Duong et al. 2015)



Selective Parameter Adaptation

- Adapt subset of parameters
- Fine-tune parts of the parameters on a low resource language (Zoph et al. 2016)

Setting	Dev BLEU	Dev PPL
No retraining	0.0	112.6
Retrain source embeddings	7.7	24.7
+ source RNN	11.8	17.0
+ target RNN	14.2	14.5
+ target attention	**15.0**	13.9
+ target input embeddings	14.7	**13.8**
+ target output embeddings	13.7	14.4

Regularization Methods for Adaptation (e.g. Barone et al 2017)

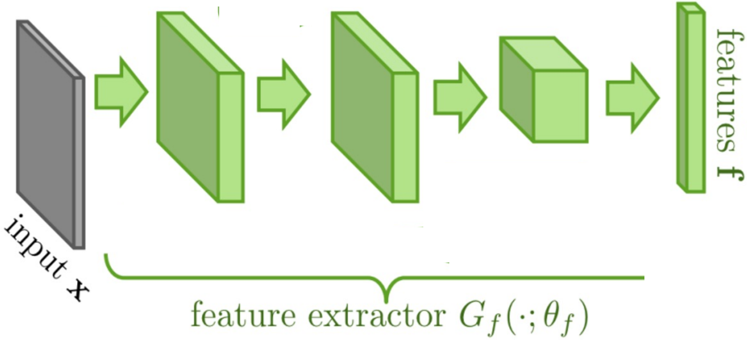
- Prevent the model from moving too far from initial value
- Regularization
  - Implicit Regularization:
    - Early Stopping - stops when model starts to overfit
    - Dropout
  - Explicit Regularization: L2 norm on difference from initial parameters

$$\theta_{adapt} = \theta_{pre} + \theta_{diff}$$

$$\ell(\theta_{adapt}) = \sum_{\langle X,Y\rangle \in \langle \mathcal{X},\mathcal{Y}\rangle} -\log P(Y \mid X; \theta_{adapt}) + ||\theta_{diff}||$$

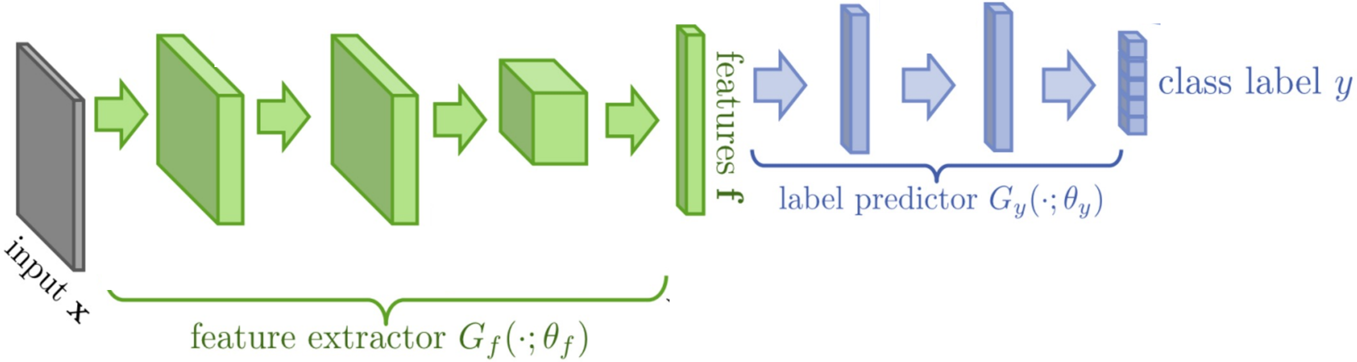
Feature Space Regularization

- Regularize features from different domains (Ganin et al 2016)
- Adversarial training



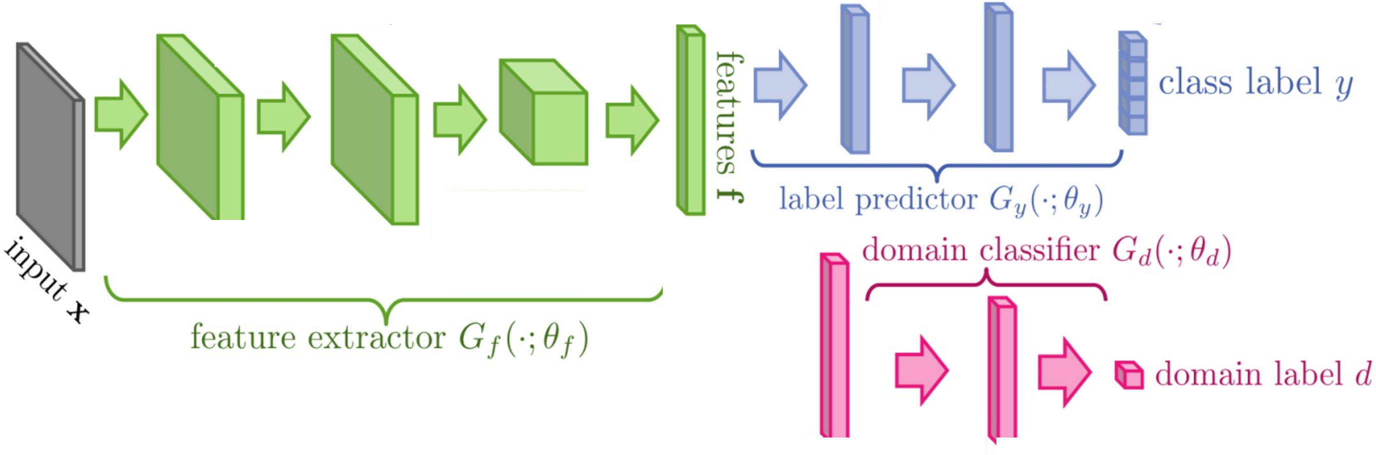
Feature Space Regularization

- Regularize features from different domains (Ganin et al 2016)
- Adversarial training



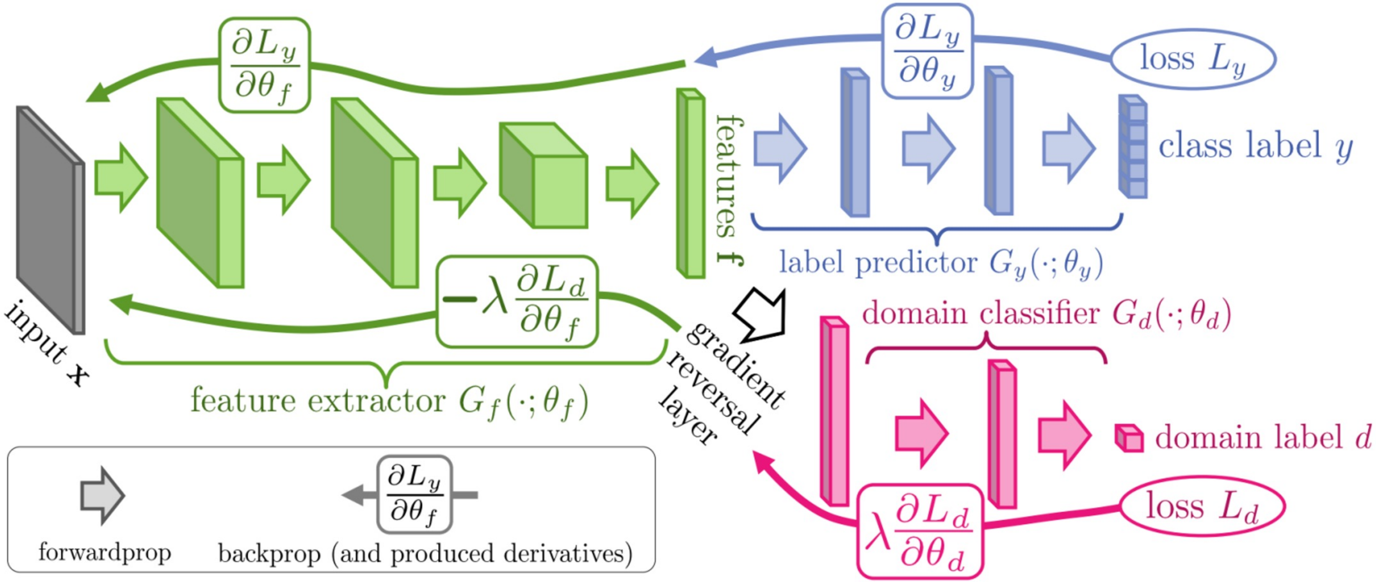
Feature Space Regularization

- Regularize features from different domains (Ganin et al 2016)
- Adversarial training



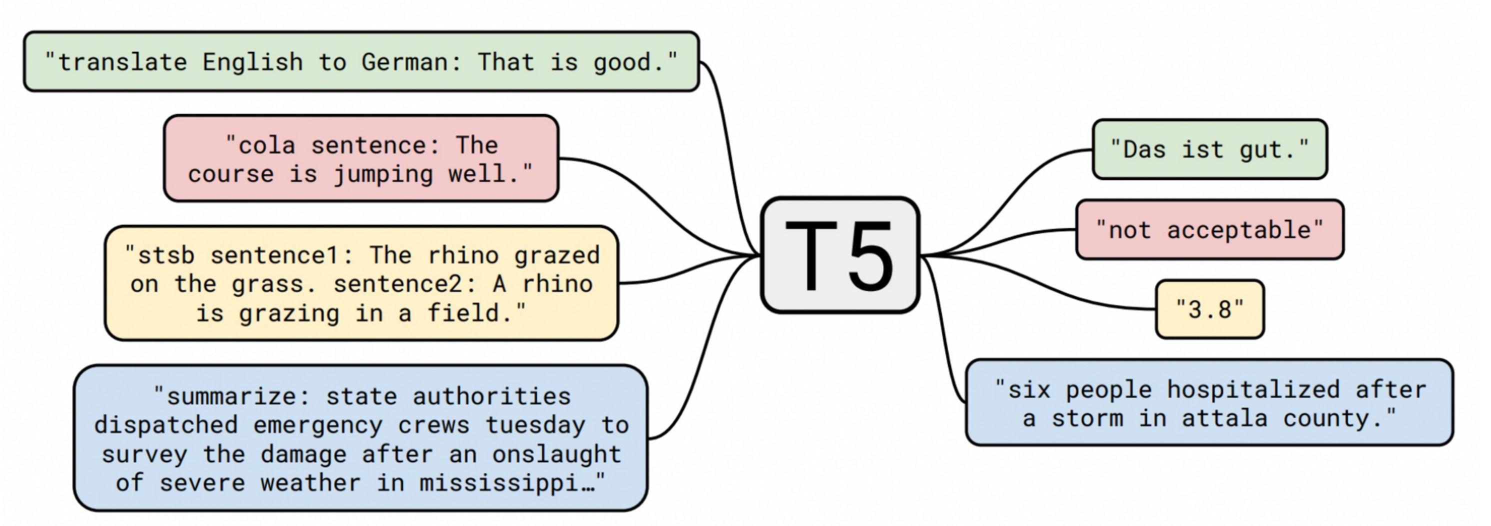
Feature Space Regularization

- Regularize features from different domains (Ganin et al 2016)
- Adversarial training



Multi-task Pre-training of Generative Models (T5)

- Convert all NLP tasks into a seq-to-seq learning format
- Append an instruction (e.g., “translate English to German”) before the real input sentence.



Raffel. et al 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Multi-tasking methods

- (Model) Parameter-sharing / Invariant Feature Learning
- **(Learning) Task Re-weighting**
- (Data) Data augmentation

Handling Different Tasks in Learning

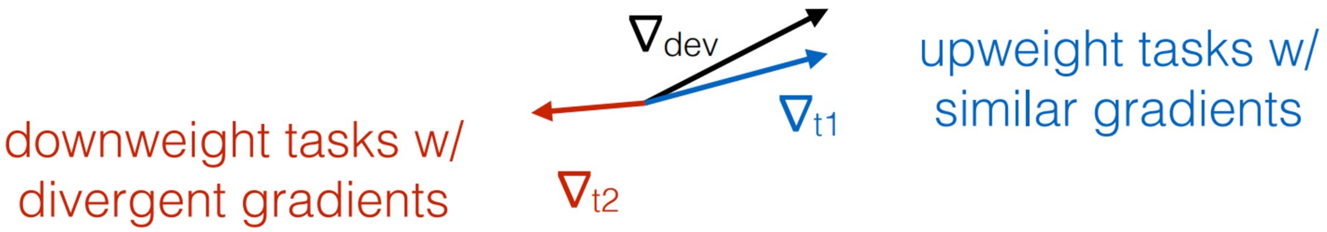
- How much to learn on each task?
  - **Task Weighting**: Differently weight loss functions from different tasks
  - **Task Sampling**: Similar to weighting, modify sampling proportion of examples from each task
- When to learn on each task?
  - Jointly vs Sequentially
  - **Curriculum Learning**: Choose the ordering of tasks with respect to difficulties

Simple Task Weighting Strategies

- **Uniform**: Sample/weight all tasks with equal probability
- **Proportional**: Sample/weight tasks according to data size
- **Temperature-based**: Sample tasks according to data size exponentiated by $1/\tau$ (Arivazhagan et al. 2019)

Data-driven Weighting Strategies

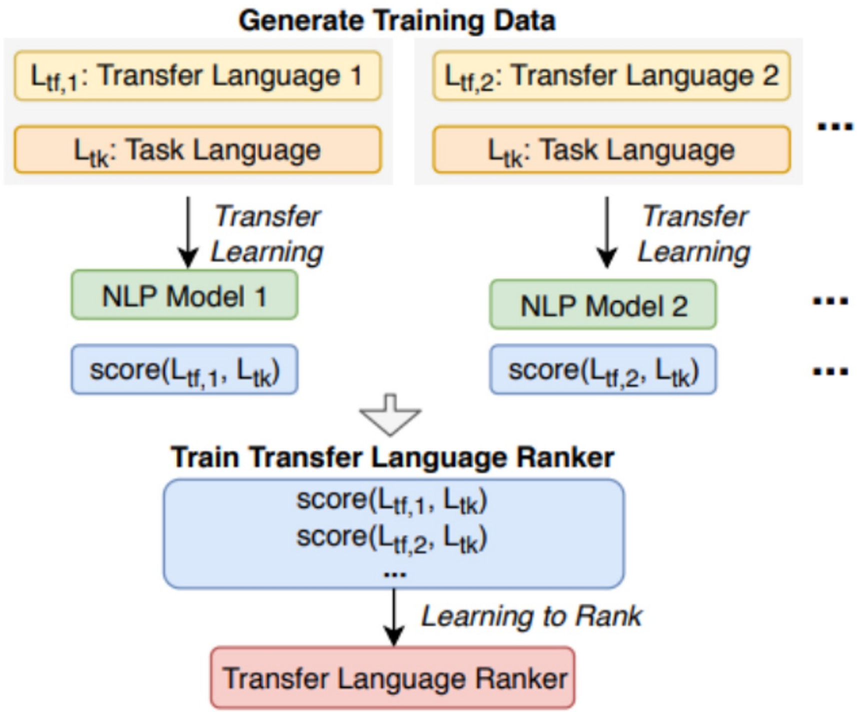
- **Loss Scaling**: For each task, estimate the variance of the neural network output assuming the output follows a gaussian distribution. Scale the loss according to variance w/ regularizer (Kendall et al. 2018)
- **Task Weight Optimization**: Optimize weights of each task to improve accuracy on a development set (e.g. Dery et al. 2021)



Choosing Transfer Tasks

We have many tasks that we could be choosing from

- **Intuitive selection**: more similar task benefit more
- **Empirical selection**: run many transfer experiments and deduce rules
  - Choosing transfer languages (Lin et al. 2019)
  - Multi-task learning on one language (Vu et al. 2020)



Distributionally Robust Optimization

Distributionally robust optimization optimizes the worst-case loss (loss on the worst task)

$$\mathcal{L} = \underset{\theta}{\operatorname{argmin}} \max_{\tilde{\mathcal{L}}} \tilde{\mathcal{L}}(\theta)$$

Improves robustness

Multi-tasking methods

- (Model) Parameter-sharing / Invariant Feature Learning
- (Learning) Task Re-weighting
- **(Data) Data augmentation**

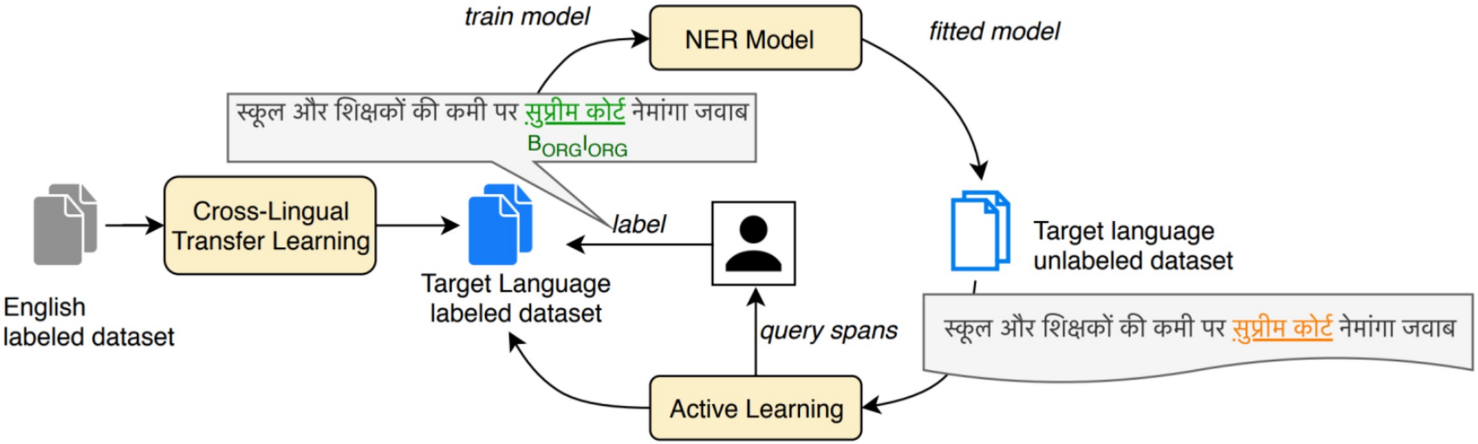
Pivot-based Methods

Use a **machine translation model / bilingual dictionary** to translate between languages

- **Translate-train**:
  - Give a labeled source dataset
  - translate the source texts into target language (with label from source)
  - train a target-language model, and predict the target test data.
- **Translate-test**:
  - Train a source-language model on the labeled source dataset.
  - Use MT to translate target data into source language.
  - Use the source-language model for prediction.

Combined with Active Learning

- Active Learning (AL) aims to select ‘useful’ data for human annotation which maximizes end model performance
- [Chaudhary et al, 2019] propose a recipe combining transfer learning with active learning for low-resource NER.



Questions?