Domain Adaptation

DSCC/LING 251/451: Machine Learning with Limited Data

C.M. Downey

Spring 2026

Roadmap

- Last time: the transfer learning taxonomy (three axes, major paradigms)
- Today: zoom into **domain adaptation** — same task, different domain
- Three parts:
    i. **The adaptation funnel** — the simplest, most practical DA strategy
    ii. **Representation alignment** — making domains look the same in feature space
    iii. **What the theory tells us** — when adaptation can and can't work

Recall: domains and tasks

- From last time (Pan & Yang, 2010):
  - Domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$: feature space + data distribution
  - Task $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$: label space + labeling function
- Domain adaptation: $\mathcal{T}_S = \mathcal{T}_T$ but $\mathcal{D}_S \neq \mathcal{D}_T$
  - Same problem, different data distribution
- Today: what exactly does $\mathcal{D}_S \neq \mathcal{D}_T$ mean, and what can we do about it?

Covariate shift: the simplest case

- If $\mathcal{T}_S = \mathcal{T}_T$, then $P_S(Y|X) = P_T(Y|X)$: the labeling function is the same
- The only thing that changes is $P(X)$ — the input distribution
- This is **covariate shift**
- Example: spam filter trained on 2020 emails, deployed in 2026
  - What counts as spam is the same; the emails look different
- This is the optimistic case — adaptation is "just" about bridging the input gap
- When it breaks down:
  - "Unpredictable" is positive for a thriller, negative for a car: $P(Y|X)$ changes too
  - Cross-lingual: surface-form-to-label mapping is completely different
  - In these cases, the "same task" assumption is strained

Examples of domain shift

- **Sentiment analysis**: product reviews $\rightarrow$ movie reviews
  - Same labels (positive/negative), but vocabulary shifts
  - "Lightweight" is positive for a laptop, irrelevant for a movie
  - Roughly covariate shift — but some features change meaning across domains
- **Medical imaging**: Hospital A's scanner $\rightarrow$ Hospital B's scanner
  - Same task (detect tumors), different pixel distributions
  - Close to pure covariate shift — the labels genuinely don't change
- **Cross-lingual NLP**: English NER $\rightarrow$ Yoruba NER
  - Same task, but the "domain" is an entirely different language
  - Covariate shift is a rough approximation at best

Which of these seems hardest? Why?

The Adaptation Funnel

Gururangan et al. (2020)

The simplest domain adaptation

- Take a pre-trained model
- Continue pre-training on **unlabeled data from your target domain**
- Then fine-tune on labeled task data
- This is **domain-adaptive pre-training (DAPT)**
- Why it works:
  - Pre-trained representations are general-purpose but not optimized for your domain
  - Continued pre-training shifts them toward the target distribution
  - Without losing general knowledge (the base is preserved)
- Simple, requires only unlabeled target-domain text
- Often a surprisingly strong baseline that more complex methods struggle to beat

The funnel: progressive domain specialization

Pre-training as a funnel from most general to most specific:

1. **General pre-training** — massive, diverse corpus (web text, Wikipedia, books)
    - Learns general-purpose linguistic / perceptual features
2. **Domain-adaptive pre-training (DAPT)** — unlabeled text from your domain
    - e.g., biomedical papers, legal documents, Gothic Bible fragments
    - Shifts representations toward the target domain
3. **Task-adaptive pre-training (TAPT)** — unlabeled text *similar to your task data*
    - Even narrower — adapts to the specific distribution you'll be evaluated on
    - Even the unlabeled portion of your task dataset counts
4. **Supervised fine-tuning** — labeled task data
    - The final, most specific stage

Key findings (Gururangan et al., 2020)

- DAPT helps. TAPT helps. They **stack**: DAPT + TAPT > either alone.
- DAPT matters more when the target domain is far from the pre-training domain
  - Biomedical and CS papers benefit more than news (news is well-represented in general pre-training)
- Even a small amount of TAPT data provides a measurable boost
- **Curated TAPT**: retrieve additional task-relevant unlabeled data $\longrightarrow$ helps even more
- This generalizes beyond text NLP:
  - Vision: ImageNet $\longrightarrow$ medical image corpus $\longrightarrow$ tumor detection
  - Speech: multilingual wav2vec $\longrightarrow$ target-language audio $\longrightarrow$ ASR
- For your projects: even with off-the-shelf models, an intermediate continued-pre-training step on domain-relevant unlabeled data is worth trying

Representation Alignment

Multilingual models as implicit alignment

- Multilingual pre-training (mBERT, XLM-R):
  - Train a single transformer on 100+ languages
  - Fine-tune on English NER $\rightarrow$ apply to other languages zero-shot
  - Shared parameterization *forces* language-general representations
- Through the DA lens:
  - Each language is a "domain" with its own $P(X)$
  - The shared encoder implicitly reduces cross-lingual divergence
  - This is domain-invariant representation learning, without an explicit alignment objective

Don't take the "shared space" story too literally

- Multilingual models are massively overparameterized
- There's evidence that different languages may activate partially distinct sub-networks
  - Lottery ticket hypothesis perspective: the "winning tickets" may differ by language
- The representations can diverge more than zero-shot results might suggest
  - The model may be doing something more complex than one clean shared space
- This is an active research question, not settled science
- When implicit sharing isn't enough, we need **explicit alignment**

Explicit alignment: various options

When implicit sharing isn't enough — especially for distant language pairs or independently trained representations:

A variety of alignment techniques exist of varying complexity:

1. **Orthogonal Procrustes** (Xing et al., 2015) — align with a bilingual dictionary
2. **MUSE** (Conneau et al., 2018) — adversarial alignment, no dictionary needed
3. **LASER** (Artetxe & Schwenk, 2019) — sentence-level alignment via parallel data
4. **Cycle-consistency** (Tien & Steinert-Threlkeld, 2022) — enforce reversible mappings between spaces

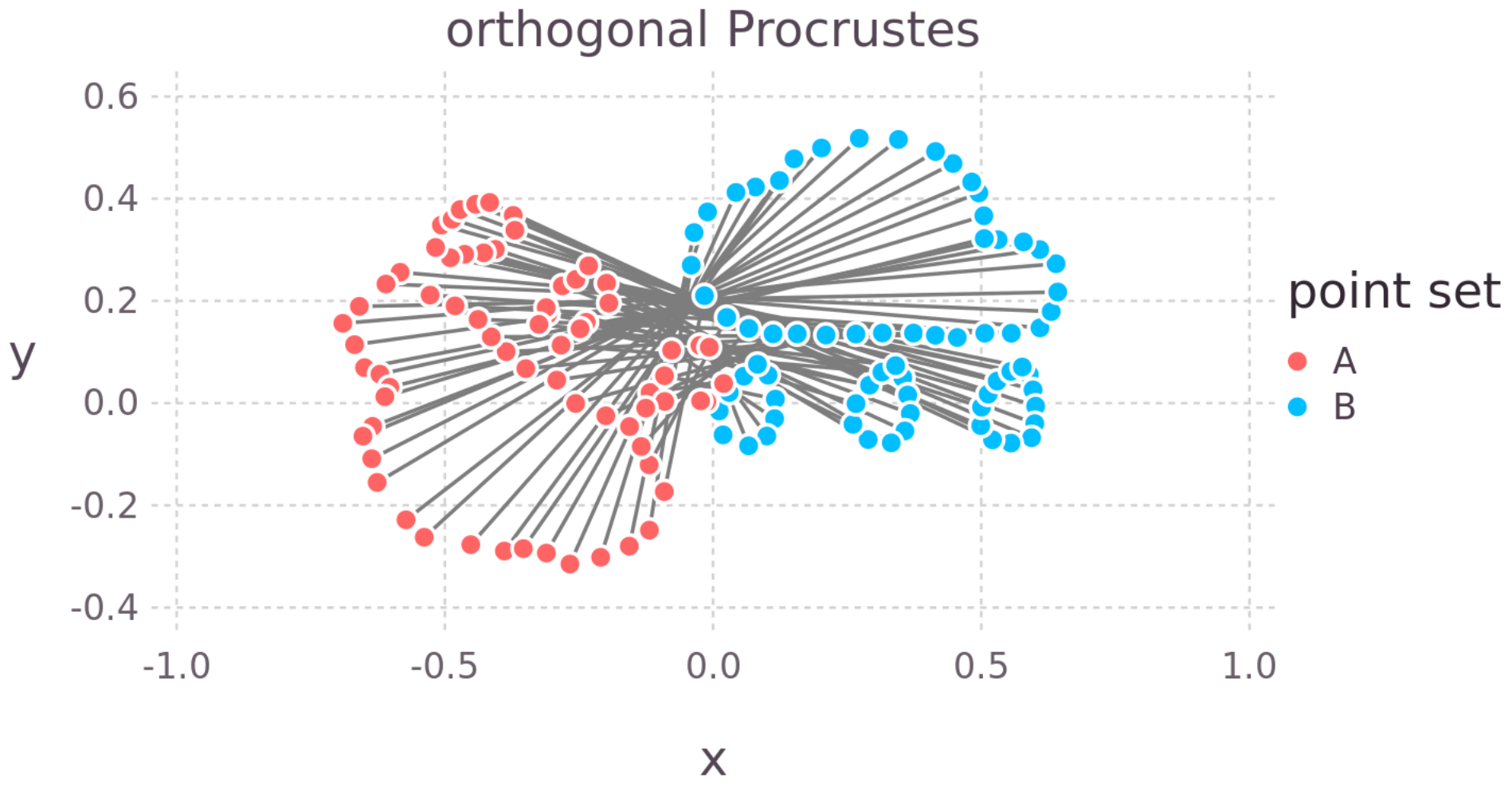
Each method relaxes the supervision requirement or changes the alignment granularity.

Orthogonal Procrustes

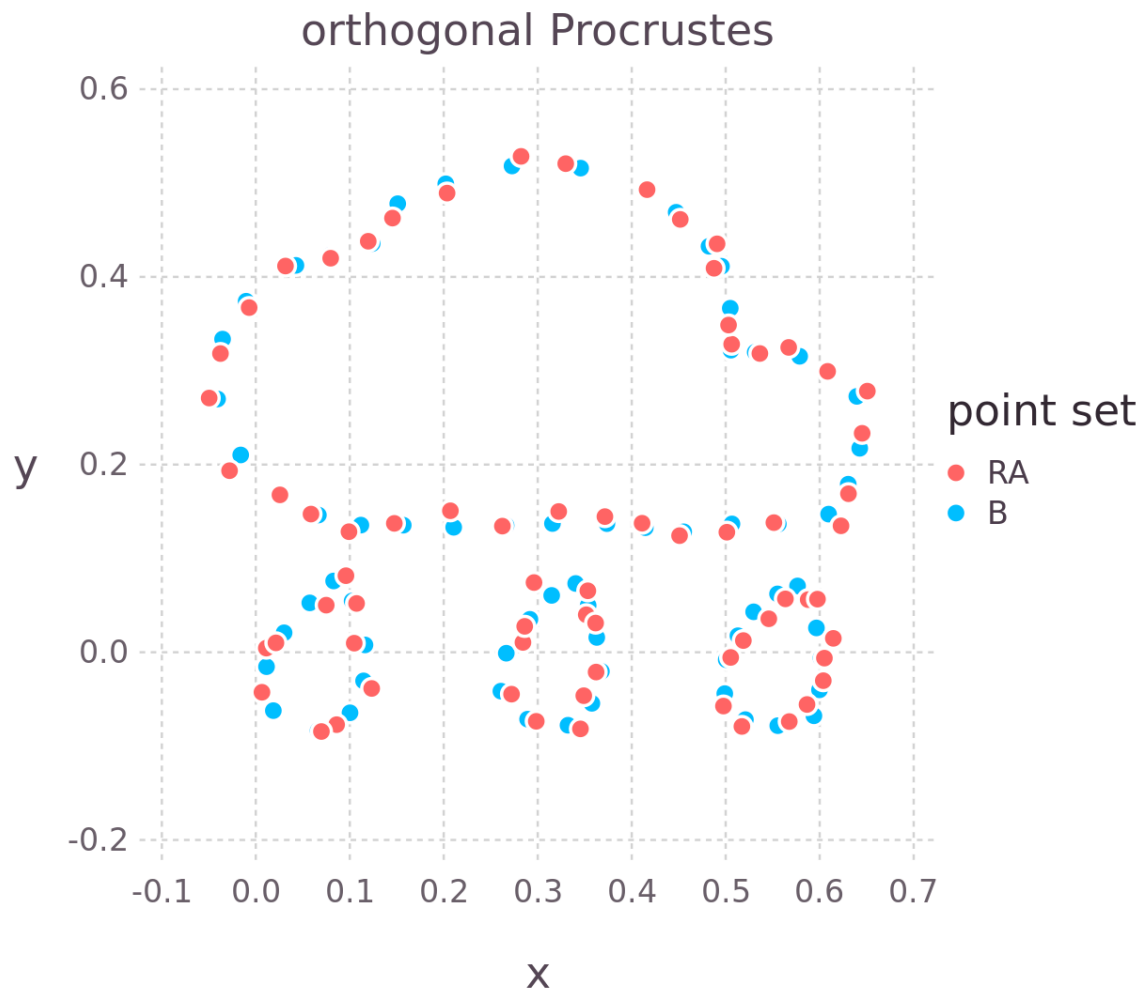
- Two independently trained monolingual word embedding spaces
- Given a bilingual dictionary (~5K word pairs), find the rotation that aligns them
- Constrain the mapping to be **orthogonal** (preserves distances within each space):

$$W^* = \arg\min_W \|WX - Y\|_F \quad \text{s.t.} \quad W^T W = I$$

- Solved in closed form via SVD
- Simple, clean, strong baseline
- Limitation: requires a bilingual dictionary as supervision



Source: Simon Ensemble blog



Source: Simon Ensemble blog

MUSE — learned mapping + adversarial alignment

What if you don't have a bilingual dictionary? (Conneau et al., 2018)

Two-stage approach:

1. **Adversarial initialization:**
   - Learn a mapping W so that a discriminator can't tell whether an embedding came from the source or target language
   - Adversarial game: mapper tries to fool the discriminator, discriminator tries to identify the language
2. **Procrustes refinement:**
   - Use the adversarial mapping to induce a synthetic dictionary (nearest neighbors)
   - Refine with Procrustes on the synthetic dictionary

Removes the need for bilingual supervision entirely.

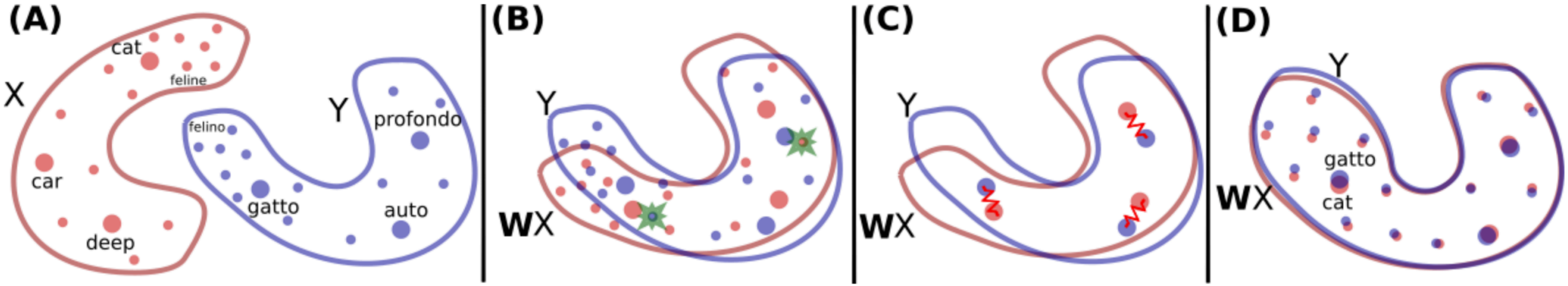
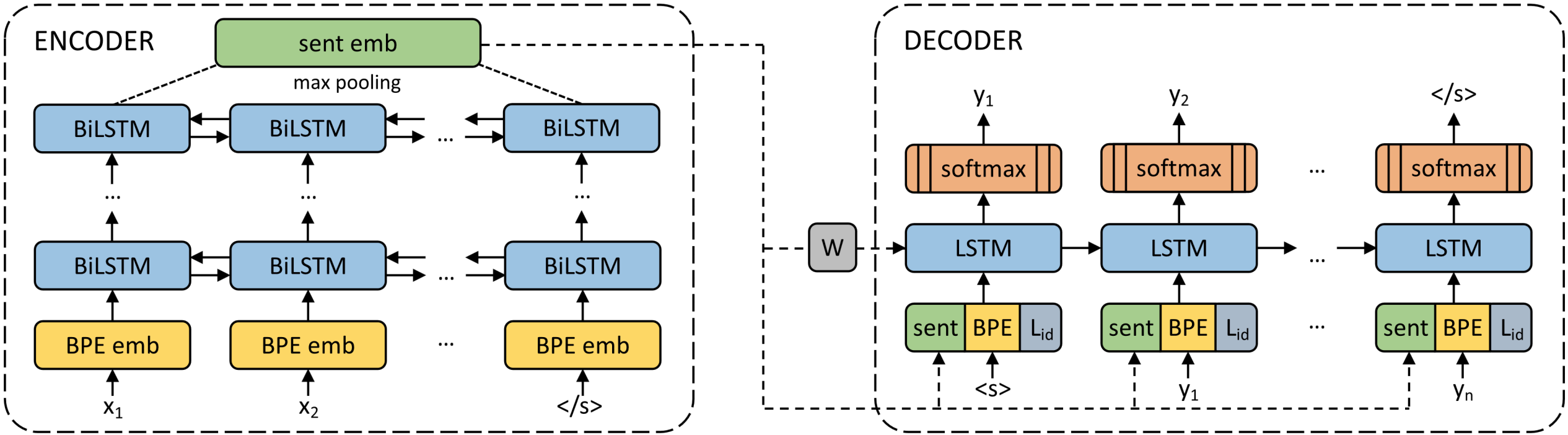


Figure 1: Toy illustration of the method. **(A)** There are two distributions of word embeddings, English words in red denoted by X and Italian words in blue denoted by Y, which we want to align/translate. Each dot represents a word in that space. The size of the dot is proportional to the frequency of the words in the training corpus of that language. **(B)** Using adversarial learning, we learn a rotation matrix W which roughly aligns the two distributions. The green stars are randomly selected words that are fed to the discriminator to determine whether the two word embeddings come from the same distribution. **(C)** The mapping W is further refined via Procrustes. This method uses frequent words aligned by the previous step as anchor points, and minimizes an energy function that corresponds to a spring system between anchor points. The refined mapping is then used to map all words in the dictionary. **(D)** Finally, we translate by using the mapping W and a distance metric, dubbed CSLS, that expands the space where there is high density of points (like the area around the word "cat"), so that "hubs" (like the word "cat") become less close to other word vectors than they would otherwise (compare to the same region in panel (A)).

LASER — sentence-level alignment

- Move from word-level to **sentence-level** alignment (Artetxe & Schwenk, 2019)
- Train a shared encoder on parallel sentences across many languages
- The encoder learns language-agnostic sentence representations
- Stronger supervision (parallel data), but parallel data is expensive
- Advantage: captures compositional meaning, not just word-level correspondence

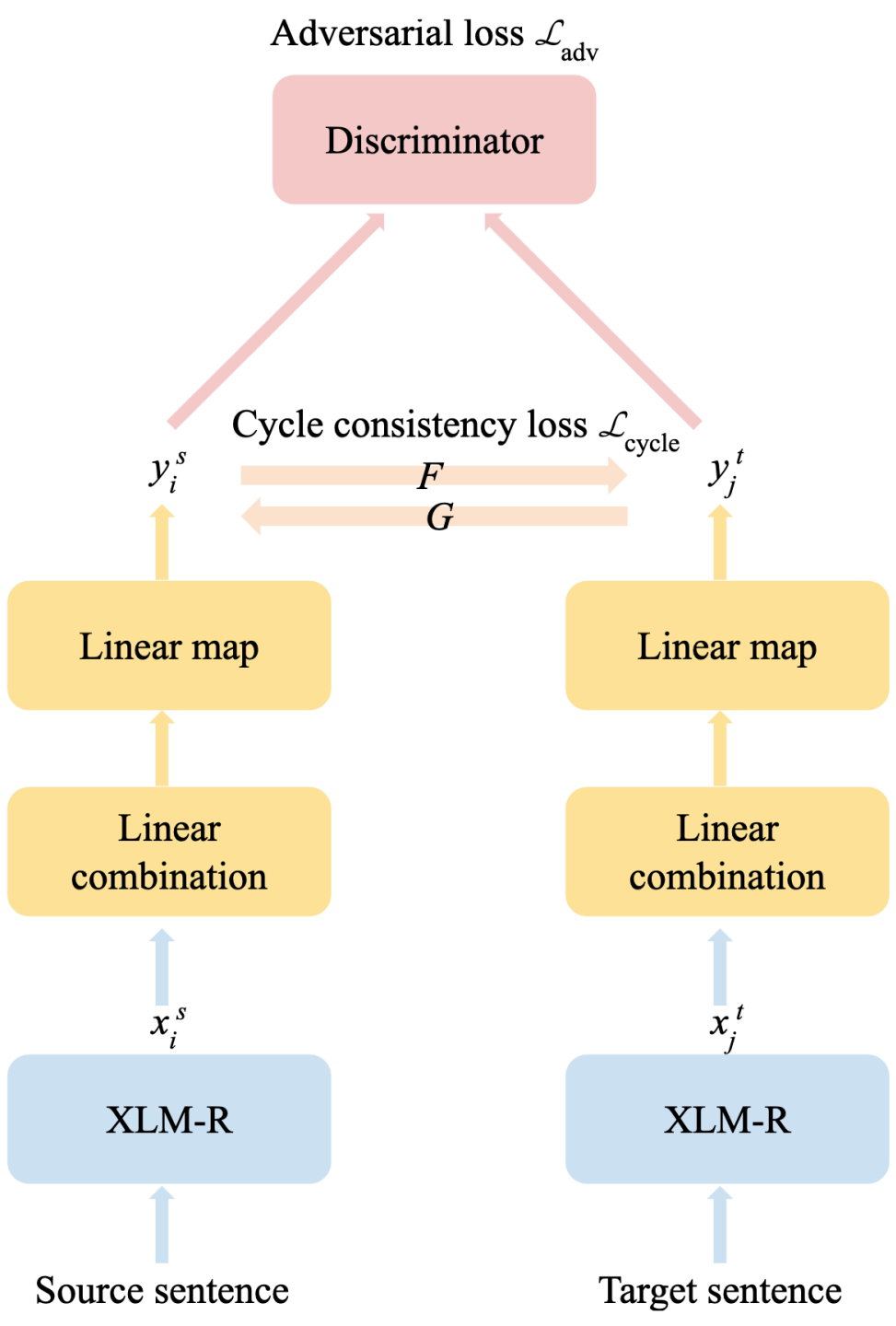

ENCODER
sent emb
max pooling
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BiLSTM
BPE emb
BPE emb
BPE emb
x1
x2
</s>
W
DECODER
y1
y2
</s>
softmax
softmax
softmax
LSTM
LSTM
LSTM
sent
BPE
Lid
sent
BPE
Lid
sent
BPE
Lid
<s>
y1
yn

Cycle-consistency loss

- Idea (Zhu et al., 2017, originally for image translation): if you map A $\longrightarrow$ B, then B $\longrightarrow$ A, you should get back where you started

$$M_{B \to A}(M_{A \to B}(x)) \approx x$$

- Encourages roughly reversible, structure-preserving mappings between spaces
- Doesn't require parallel data — only internal consistency of the mapping
- Tien & Steinert-Threlkeld (2022) apply this to **sentence embedding alignment**, combined with adversarial training:
  - Adversarial loss: encoder outputs should be indistinguishable across languages
  - Cycle-consistency loss: round-trip through both languages should reconstruct the original


Adversarial loss $\mathcal{L}_{adv}$
Discriminator
Cycle consistency loss $\mathcal{L}_{cycle}$
y_i^s
F
G
y_j^t
Linear map
Linear map
Linear combination
Linear combination
x_i^s
x_j^t
XLM-R
XLM-R
Source sentence
Target sentence

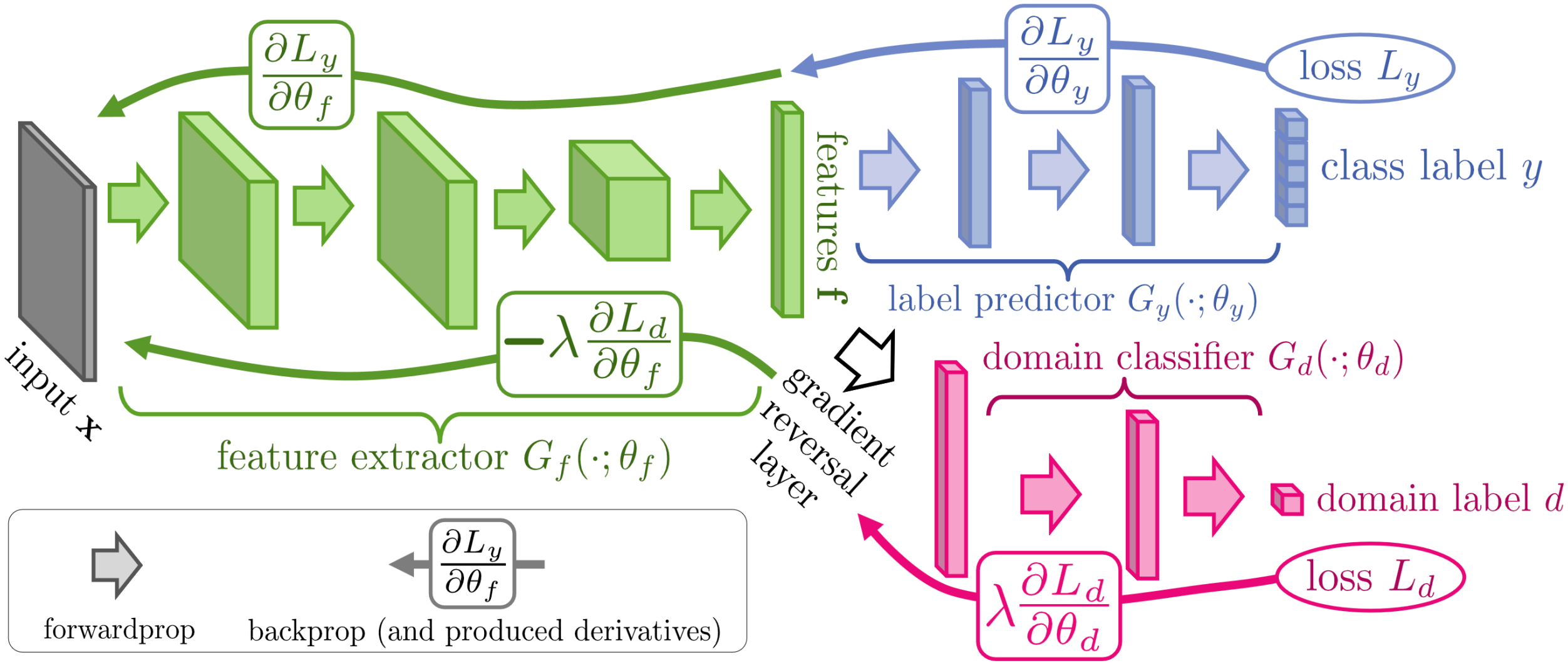
The invariance-discriminability tension

- All alignment methods face a fundamental tradeoff:
  - We want features where source and target **look the same** (domain-invariant)
  - But aggressive alignment can **destroy task-relevant information**
- Perfect invariance is trivial: map everything to a constant
  - Zero divergence, zero predictive power
- Goal: features that are **invariant to domain** but **discriminative for the task**
- The adversarial framework handles this explicitly...

DANN: the general adversarial framework

Domain-Adversarial Neural Networks (Ganin et al., 2016):

- **Shared encoder** produces representations from both domains
- **Task classifier**: predict labels (trained on source labels)
- **Domain discriminator**: predict source vs. target (trained on both)
- Encoder is trained to **help the task classifier** while **fooling the domain discriminator**
- Intuition: learn features that are useful for the task but don't reveal which domain they came from
- How it connects to what we've seen:
  - MUSE's adversarial stage *is* DANN for word embeddings
  - Multilingual pre-training does something similar implicitly
  - General principle: use a domain discriminator as a training signal for alignment


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
loss L_y
class label y
features $\mathbf{f}$
label predictor $G_y(\cdot;\theta_y)$
$-\lambda\frac{\partial L_d}{\partial \theta_f}$
gradient reversal layer
domain classifier $G_d(\cdot;\theta_d)$
input $\mathbf{x}$
feature extractor $G_f(\cdot;\theta_f)$
domain label d
loss L_d
$\lambda\frac{\partial L_d}{\partial \theta_d}$
forwardprop
$\frac{\partial L_y}{\partial \theta_f}$
backprop (and produced derivatives)

Other Adaptation Methods

(Brief tour)

Importance weighting

- Under covariate shift, the target risk can be rewritten as an expectation over the **source**:

$$\epsilon_T(h) = \mathbb{E}_{x \sim P_T}[\ell(h(x), y)] = \mathbb{E}_{x \sim P_S}\left[\frac{P_T(x)}{P_S(x)}\ell(h(x), y)\right]$$

- So training on source with weights $w(x) = P_T(x)/P_S(x)$ is exactly equivalent to training on target
- Practically hard: estimating density ratios in high dimensions is fragile; extreme weights → high variance
- **KMM** (Huang et al., 2006): sidestep density estimation — find weights that match the mean of source and target in a kernel space

Data selection

- Simpler alternative to reweighting: just **select** the most relevant source data
- Applicable when you have a large, heterogeneous source pool
  - Example: building an LM for a low-resource language from a multilingual corpus → select the most similar subset
- Moore & Lewis (2010): score source sentences by in-domain vs. out-of-domain LM probability ratio — high ratio means relevant to target
- Simple, effective, underrated
- Connects to the funnel: data selection is choosing what goes into the earlier stages

Self-training under domain shift

- Self-training (Lecture 8) is a natural fit for DA:
  - You have labeled source + unlabeled target — built-in semi-supervised structure
- Procedure: train on source → pseudo-label target → retrain on both → iterate
  - Filter by confidence threshold τ to reduce noise
- Risk: confident-but-wrong predictions compound (confirmation bias)
- **Gradual adaptation** (Kumar et al., 2020):
  - If the gap is too large, chain through intermediate domains
  - Cross-lingual: bridge languages (English $\longrightarrow$ Dutch $\longrightarrow$ German)
  - Each step keeps the shift small enough for self-training to work

What the Theory Tells Us

The Ben-David bound (headline version)

For hypothesis $h \in \mathcal{H}$ (Ben-David et al., 2010):

$$\epsilon_T(h) \leq \epsilon_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda^*$$

Term	Meaning	Intuition
$\epsilon_S(h)$	Source error	Do well on source first
$d_{\mathcal{H}\Delta\mathcal{H}}$	Domain divergence	How different the domains look to your model
λ^*	Adaptability	Can a single model work for both domains?

- The methods we've seen today reduce source error or domain divergence
- λ^* is the hard limit: if the task genuinely differs across domains, you're stuck
- This is the theoretical version of "negative transfer"

What is $d_{\mathcal{H}\Delta\mathcal{H}}$?

- Domain distance that's relative to **your model class**, not just statistics
- $\mathcal{H}\Delta\mathcal{H}$: the set of regions where two hypotheses $h, h' \in \mathcal{H}$ disagree
- The divergence: is there a disagreement region with very different probability under source vs. target?

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) = 2 \sup_{h,h' \in \mathcal{H}} \left| \Pr_{\mathcal{D}_S}[h(x) \neq h'(x)] - \Pr_{\mathcal{D}_T}[h(x) \neq h'(x)] \right|$$

- Divergence $= 0$: domains look the same to every classifier pair you can build
- Divergence is large: some classifier pair disagrees in completely different places across domains

Proxy-$\mathcal{A}$-distance: measuring domain distance

- A practical diagnostic you can use right now:
  - Train a classifier to distinguish source examples from target examples
  - **50% accuracy** (chance): domains are hard to tell apart $\rightarrow$ good news
  - **100% accuracy**: domains are trivially separable $\rightarrow$ large gap
  - $\hat{d}_{\mathcal{A}} = 2(1 - 2\epsilon)$ gives a scalar estimate
- This is the idea behind DANN:
  - Instead of just *measuring* domain distance, use the discriminator as a *training signal* to reduce it
- Always run the no-adaptation baseline first
  - If zero-shot source model is near chance on target, ask: do I really have the "same task"?

Practical takeaways

1. **Start with the funnel**: continued pre-training on target-domain data (DAPT/TAPT) is the simplest DA strategy — try it first
2. **Representation alignment is the core principle**: implicit (multilingual models) or explicit (Procrustes, adversarial, cycle-consistency)
3. **Measure before you adapt**: proxy-$\mathcal{A}$-distance tells you how big the gap is
4. **Know when to walk away**: if zero-shot is near chance, the task may not be "the same" — a few target labels may be worth more than sophisticated unsupervised DA

Where DA fits in the course

- **Lecture 3** (Inductive bias): DA bets that source and target share structure
- **Lecture 7** (Self-supervised): the funnel starts with self-supervised pre-training
- **Lecture 8** (Semi-supervised): self-training for DA = semi-supervised with distributional shift
- **Lecture 10** (Transfer): DA is a well-studied corner of the taxonomy
- **Next time** (Few-shot): what if you have a few target labels?

For Tuesday's discussion

- Think about domain shift in **your** project:
  - Is there a distribution mismatch between your training data and your target setting?
  - Could continued pre-training on domain-relevant data help?
- When reading the discussion papers:
  - What kind of domain shift is involved?
  - Which adaptation strategy do the authors use?

References

- Gururangan et al. (2020), *Don't Stop Pretraining*
- Xing et al. (2015), *Normalized Word Embedding and Orthogonal Transform*
- Conneau et al. (2018), MUSE: *Word Translation Without Parallel Data*
- Artetxe & Schwenk (2019), LASER
- Ganin et al. (2016), *Domain-Adversarial Training of Neural Networks*
- Shimodaira (2000), *Improving Predictive Inference Under Covariate Shift*
- Huang et al. (2006), *Correcting Sample Selection Bias by Unlabeled Data*
- Moore & Lewis (2010), *Intelligent Selection of Language Model Training Data*
- Kumar et al. (2020), *Self-Training for Gradual Domain Adaptation*
- Ben-David et al. (2010), *A Theory of Learning from Different Domains*
- Ramponi & Plank (2020), *Neural Unsupervised Domain Adaptation in NLP*