Induced Model Matching: Learning from Restricted Models

Doctoral Dissertation Defense

Usama Muneeb

Personal Introduction

- Joined UIC in Fall 2017 and the DICE group in Fall 2020
  - DICE = Data, Information and Computing, Equitably
- Main thesis motivation: maximally utilizing feature-restricted models as side information when training neural models
  - We started off with *noising* as our baseline: an approach by Stanford ML to transfer (structural) Kneser–Ney bigram knowledge into an LSTM RNN,
  - We explained the principle behind noising, connected it to reverse knowledge distillation (KD), and revealed them as suboptimal methods of utilizing such side information,
  - We developed a methodology to learn from restricted models, and showed that is applicable beyond language models and is able to surpass noising and reverse KD.

Committee Members

- Professor Mesrob I. Ohannessian (Advisor and Committee Chair, UIC ECE)
- Professor Ahmet Enis Cetin (UIC ECE)
- Professor Shuo Han (UIC ECE)
- Professor Brian D. Ziebart (UIC CS)
- Professor Natalie Parde (UIC CS)

Overview

Side information and restricted models

ENGINEERING

Electrical and Computer Engineering

How do we define *side information*?

Definition

Side information about a dataset is anything known about the data but not additional data. It can also be defined as structural information.

Example 1: Convergence in probability

We want to upper bound the probability of the sum $X = \sum_i^n X_i$ of random variables distributed as $X_i \sim$ **Bernoulli**(p) is greater than a.

- e.g. the probability that we get at least $a = 10$ heads from $n = 20$ coin tosses, i.e. $\mathbb{P}(X > a)$.

What's the best bound we can get?

- The Markov's inequality gives us an upper bound of only $\frac{\mathbb{E}(X)}{a}$ on $\mathbb{P}(X > a)$.
- Chernoff bound gives us an exponentially decreasing upper bound.
  - The Chernoff bound accounts for not just the *mean* and *variance* of the data but also the **side information** that we are dealing with a *sum* (or average) of these set of numbers.

Example 2: ML estimation of a probability matrix

- We have a conditional probability matrix $P(Y = j|X = i)$
  - We compute and know its rank, which is **side information**.
- We sample counts C_{ij} from this matrix.
  - We let C_{ij} be **sparse** relative to $P(Y|X)$.
- We use a *tempered EM* algorithm to compute an ML estimate of C_{ij}.
  - It's done as part of the PLSA algorithm that performs the factorization $P(Y = y|X = x) = P(Y = y|Z = z)P(Z = z|X = x)$
  - Rank takes the role of the (length of the) latent dimension Z
- The resulting $\hat{P}(Y|X) = \hat{P}(Y|Z)\hat{P}(Z|X)$ very closely resembles $P(Y|X)$.
  - C_{ij} was sparse but the algorithm knew the rank through the latent dimension Z!

PLSA inputs and outputs visualized

C_{xy} alone cannot give us $\hat{P}(Y|X)$, but combined with its rank information, it can!

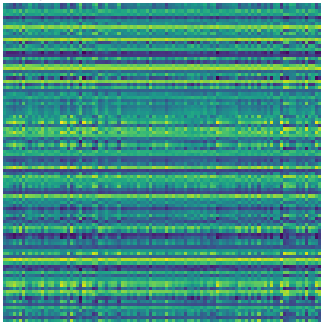


Figure: True $P(Y = y|X = x)$

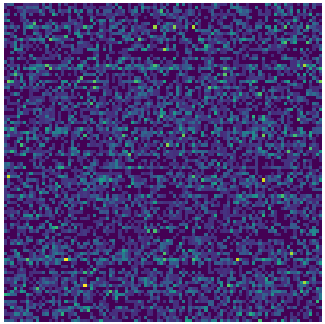


Figure: Counts $C_{xy} \sim P(Y|X)$

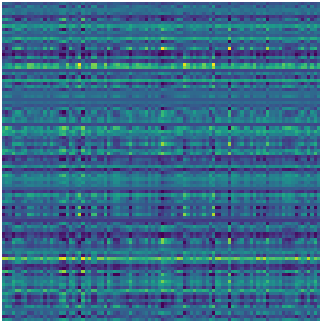


Figure: Estimated $\hat{P}(Y|X)$

Induced Models

Our Methodology

Intuition - Woodblock Printing

- A technique for replicating a pattern on paper or textile
  - Was in widespread use in China during 7th century (i.e. Tang Dynasty)
- A merchant bought silk and wanted it block printed
  - Woodblock was not fine enough fit the intricacies of their pattern prototype
  - They *folded the silk* after woodblock printing, and *refined it by hand*.
- We propose a similar technique in training neural models, where we *fold the neural model* to *refine it* using side information at a restricted scale.

Figure: Woodblock printing

Side Information in Logistic Regression

Introductory example and proof of concept

- Suppose we are training a classifier $Q(Y = 1 \mid x_1, x_2, x_3)$
- Assume we also know $\overline{P}(Y^i = 1 \mid x_1^i)$ exactly for every sample (i.e. Bayes optimal $\overline{P}$)

$$\overline{P}(Y = 1 \mid x_1) = \frac{\text{Probability mass of blue polygon}}{\text{Mass of both blue and red polygon}}$$

- $\overline{P}(y|\overline{x})$ depends only on the restricted feature set x_1 where the complete feature set is $x = (x_1, x_2, x_3)$.
  - To distinguish *restricted* and *extended* feature sets, we use $\overline{x} = x_1$ and $\underline{x} = (x_2, x_3)$

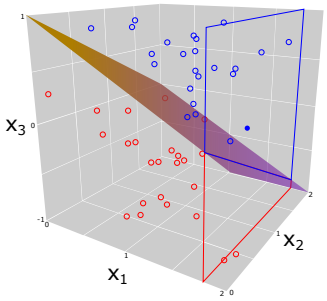


Figure: Restricted model

$\overline{P}$ is a marginal, how to use it?

- In general, $\overline{P}(y \mid x)$ is defined as a marginal of P (which isn't available in practice).

$$\overline{P}(y|\overline{x}) = \sum_{\underline{x}} \mathbf{P}(y, \underline{x}|\overline{x}) = \sum_{\underline{x}} \pi(\underline{x}|\overline{x})\, P(y|\overline{x}, \underline{x})$$

  - For now we assume we know $\overline{P}$ exactly (analytically defined as a ratio of the polygon areas).
- **Fundamental question**: $\overline{P}$ performs classification at a smaller scale than Q
  - How can we utilize this restricted scale $\overline{P}(Y = 1 \mid x_1)$ while training the full featured $Q(Y = 1 \mid x_1, x_2, x_3)$?

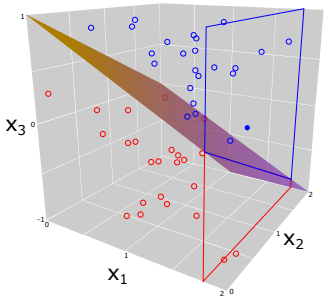


Figure: Restricted model

Existing Approaches / Related Works

- Our main baseline is **data noising** [1] which presents an approach to *noise* the inputs/outputs of a language model using the restricted model $\overline{P}$.
- We will defer the discussion about data noising to the next section (language models)
  - **Takeaway for now**: *we are not the first ones* to utilize a smaller model in the training of a larger one!
- Our approach however, does solve some caveats of noising (to be discussed later).

Reference

[1] Data noising as smoothing in neural network language models. Xie et al. *arXiv preprint arXiv:1703.02573, 2017*, 2017

Bayes optimality assumption for $\overline{P}$

- We assumed knowledge of Bayes Optimal $\overline{P}$
- Doesn't this assumption violate our definition of *side information*?
  - Yes it does!
  - Side-information should not be *more data*, which $\overline{P}(Y^i = 1 \mid x_1^i)$ is for now.
- We only use it for proof of concept, to check whether our *methodology* has merit.
  - We will remove this assumption later and replace $\overline{P}$ with a proxy, constructed from existing data.

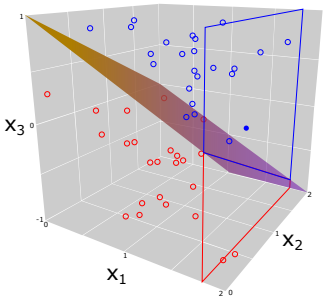


Figure: Restricted model

Learned Feature-Restricted Induced Model

a **core component** of the IMM methodology

- **We propose** that in order to learn from $\overline{P}(Y^i = 1 \mid x_1^i)$, $Q(Y = 1 \mid x_1, x_2, x_3)$ must "fold" and become $\overline{Q}(Y^i = 1 \mid x_1^i)$, just like the silk cloth did, for refinements.
- How do we do this "folding"?
  - **Answer**: simple marginalization, just as we do to (theoretically) get $\overline{P}$ from P.
- With $\overline{x} = x_1$ and $\underline{x} = (x_2, x_3)$, we have

$$\overline{Q}(y|\overline{x}) = \sum_{\underline{x}} \underbrace{\pi(\underline{x}|\overline{x})}_{\text{true context distribution}} Q(y|\overline{x}, \underline{x})$$

Context Distribution

- Just like $\overline{P}$, the true context distribution π is also a result of the marginalization of P

$$\overline{P}(y|\overline{x}) = \sum_{\underline{x}} \mathbf{P}(y, \underline{x}|\overline{x}) = \sum_{\underline{x}} \pi(\underline{x}|\overline{x})\, P(y|\overline{x}, \underline{x})$$

  - and therefore, not available in practice!
- We have ways to approximate it.
  - Approximation depends on the kind of dataset.
  - We will discuss individually for each case.

Induced Model Matching

- At this stage, we are ready to introduce the **idealized** *induced model matching* (IMM) risk

$$\sum_x \underbrace{\pi(x)}_{\text{true context distribution}} \underbrace{\sum_y \overline{P}(y|\overline{x}) \log \frac{\overline{P}(y|\overline{x})}{\overline{Q}(y|\overline{x})}}_{\mathsf{D}(\overline{P}(\cdot|\overline{x})\|\overline{Q}(\cdot|\overline{x}))}$$

  — i.e., the average context-conditional KL divergence with the restricted context ($\overline{x}$)

Empirical Feature-Restricted Induced Models

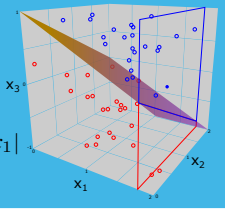
a **core component** of the IMM methodology

The process of induction is heavily reliant on the *type of dataset* in use.

For continuous datasets

Since we understood $\pi(\underline{x}|\overline{x})$ can be approximated as a soft nearest-neighbor density estimate $f(\underline{x}|\overline{x}) = f(x_2, x_3|x_1) \propto \sum_{t=1}^{n} \delta_{x_{2,t},x_{3,t}}(x_2, x_3) \mathrm{e}^{-\alpha|x_{1,t}-x_1|}$, where $1/\alpha$ is the bandwidth of the kernel. Then,

$$\begin{aligned}\overline{Q}(y|\overline{x}) &= \int f(\underline{x}|\overline{x}) Q(y|\overline{x}, \underline{x}) \\ &\approx \sum_{t=1}^{n} \frac{w_t(x)}{\sum_{t=1}^{n} w_t(x)} Q(y|\overline{x}, \underline{x}_t), \quad \text{where } w_t(x) = \mathrm{e}^{-\alpha|x_{1,t}-x_1|}\end{aligned}$$



Empirical IMM Risk

The **empirical IMM risk** is then defined using $\hat{P}$ and $\hat{Q}$ (empirical versions of $\overline{P}$ and $\overline{Q}$)

$$\mathrm{IMM}(Q) = \sum_x \pi_n(x) \sum_y \hat{P}(y|\overline{x}) \log \frac{1}{\hat{Q}(y|\overline{x})}$$

In order to weave the above into an existing training pipeline, we expand the empirical context distribution $\pi_n(x)$ and write $\mathrm{IMM}(Q)$ as a pass over the dataset:

$$\mathrm{IMM}(Q) = -\frac{1}{n} \sum_t \underbrace{\left[\sum_y \hat{P}(y|\overline{x}_t) \log \hat{Q}(y|\overline{x}_t) \right]}_{\mathrm{IMM}_t(Q)} .$$

Incorporating IMM into the training pipeline

$\overline{x}$	Restricted feature set
$\underline{x}$	Extended feature set
$x = (\overline{x}, \underline{x})$	Complete feature set

Table: Terminology

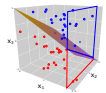


Figure: For $x = (x_1, x_2, x_3)$, $\overline{x} = x_1$ and $\underline{x} = (x_2, x_3)$

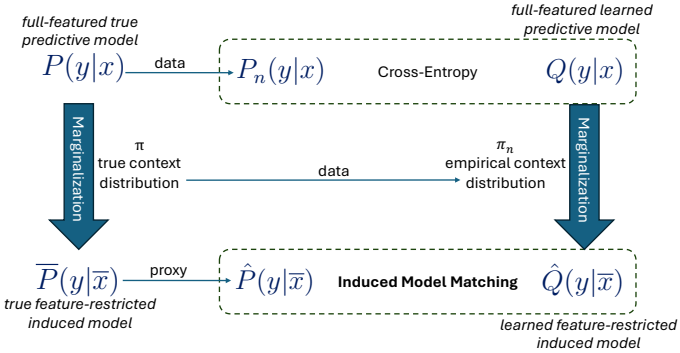


Figure: General IMM implementation, using a proxy for $\overline{P}$

IMM as a regularizer

- IMM does not replace the main objective
  - It's added as a **regularization** term to the main objective to help SGD find the parameters for the main objective.
- The main objective is still training the neural model on the full-featured task! The problem can be written as

$$\min \textbf{Cross-Entropy}(Q) + \lambda \textbf{IMM-Risk}(\overline{P}, \hat{Q})$$

- Which later will become (after replacing $\overline{P}$ with a proxy)

$$\min \textbf{Cross-Entropy}(Q) + \lambda \textbf{IMM-Risk}(\hat{P}, \hat{Q})$$

Performance on restricted task

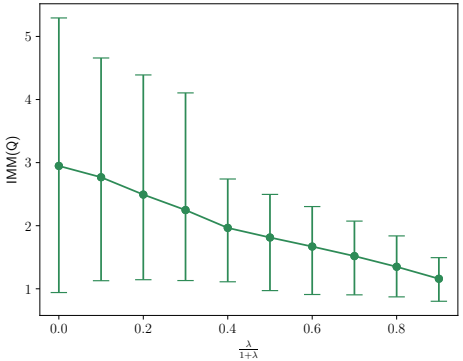


Figure: Performance of feature-restricted model on restricted task

Experimental Results on Logistic Regression

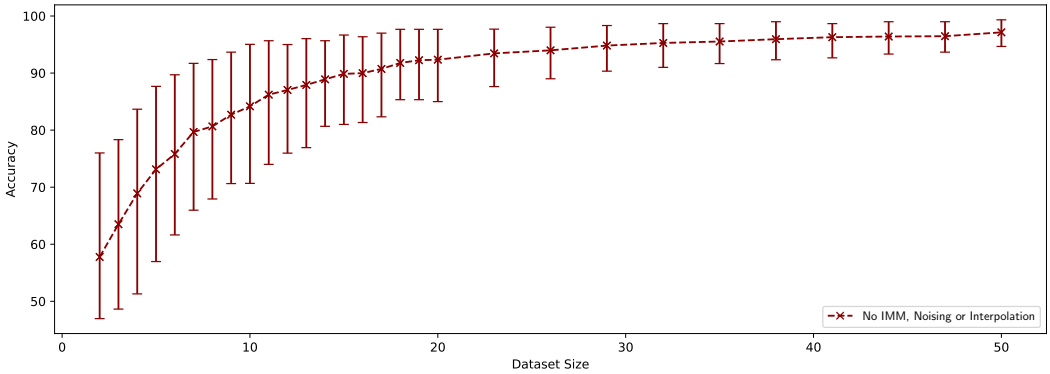


Figure: Performance of IMM in comparison to baselines

Experimental Results on Logistic Regression

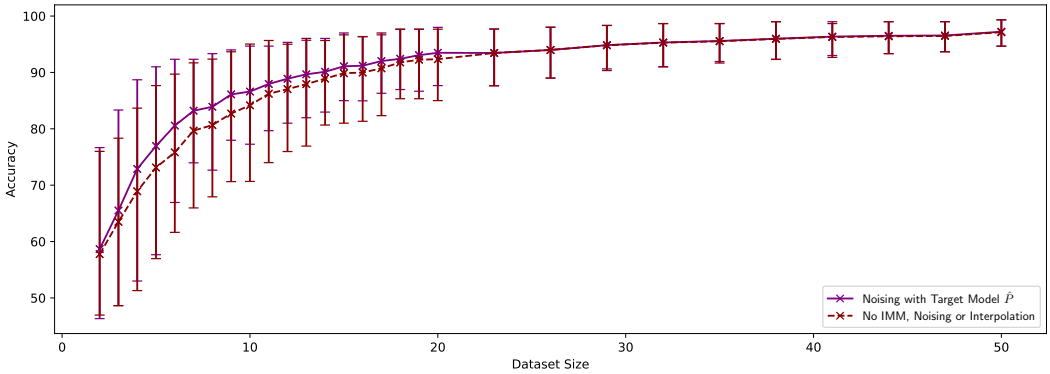


Figure: Performance of IMM in comparison to baselines

Experimental Results on Logistic Regression

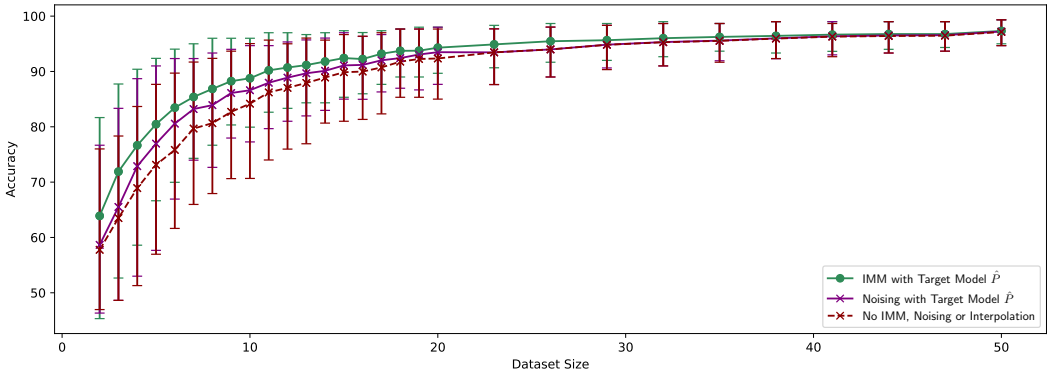


Figure: Performance of IMM in comparison to baselines

Experimental Results on Logistic Regression

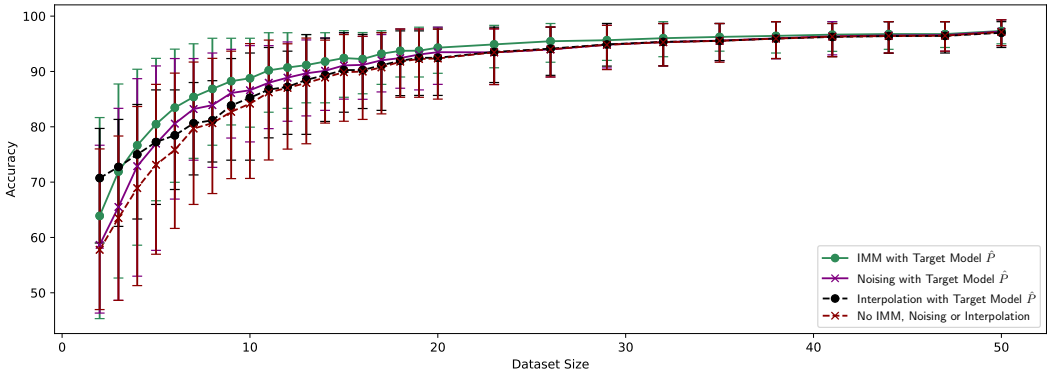


Figure: Performance of IMM in comparison to baselines

What if the target has lower quality than $\overline{P}$?

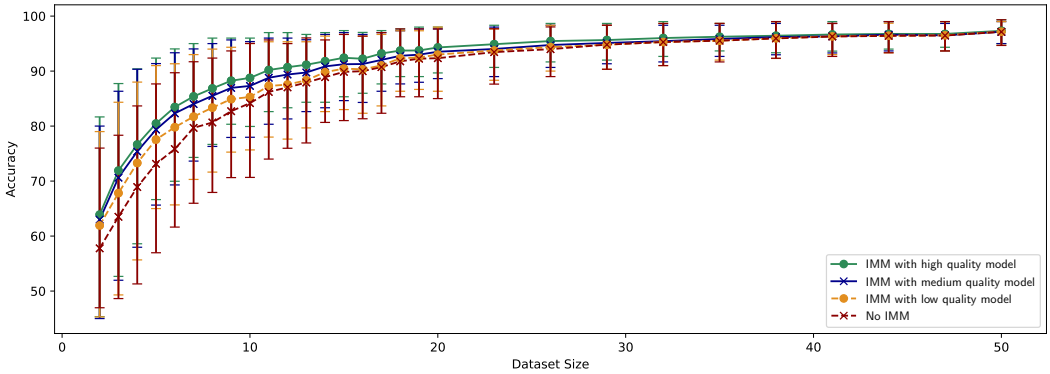


Figure: Performance of IMM with models of decaying quality

Language Modeling

From Noising to IMM

ENGINEERING

Electrical and Computer Engineering

$\hat{P}$, a proxy for $\overline{P}$

- In the case of Logistic Regression, we had obtained it using the probability masses of the trapezoids.
  - That is basically marginalizing the generating distribution P from which we sample the data!
- Since we don't have P, we simply *cannot* have $\overline{P}$
  - It's time to remove assumption about *any* knowledge of $\overline{P}$
- Fortunately, we can construct a proxy for $\overline{P}$ from data!
  - Let's talk about a proxy $\hat{P}$ for $\overline{P}$

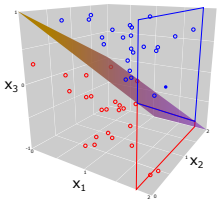


Figure: Restricted model in Logistic Regression (earlier)

$\hat{P}$, a proxy for $\overline{P}$

- **A legitimate question**: why would that even help? Does not the main loss also use data?
  - In restricted dimensionality, we often can construct more informative models than in the full dimensionality space.
    - In language modeling, we can construct the famed **Kneser-Ney bigram**!
  - The knowledge of the larger model *is not a superset* of the knowledge of structurally generated smaller models.
    - Smaller models constructed using structured techniques still know something the larger neural model does not know!

The Kneser–Ney bigram

- A *smoothed* bigram that takes structural information into account (instead of just occurence counts)
- This is a strength of Kneser–Ney that helped it remain unbeaten by LSTM RNN based models for a long time. [2] [3]

References

[2] Chen, S. F., & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4), 359–394.
[3] Chelba, C., Norouzi, M., & Bengio, S. (2017). N-gram language modeling using recurrent neural network estimation. arXiv Preprint arXiv:1703. 10724.

Learned Feature-Restricted Induced Model

a **core component** of the IMM methodology

- As discussed earlier, it's obtained through marginalization.

$$\overline{Q}(y|\overline{x}) = \sum_{\underline{x}} \underbrace{\pi(\underline{x}|\overline{x})}_{\text{true context distribution}} \quad Q(y|\overline{x},\underline{x})$$

- In this case, we have $\overline{x}$ and $\underline{x}$ as the **short and extended contexts**.
- What we have obtained is an *induced bigram* of the full-featured n-gram language model (a 36-gram LSTM RNN in our case)!
  - n-gram = model that performs prediction using the complete context, i.e., x
  - induced bigram = model that performs prediction solely on the basis of the most recent token of the context, i.e., $\overline{x}$

Empirical Feature-Restricted Induced Models

a **core component** of the IMM methodology

Recall: the context distribution $\pi(\underline{x}|\overline{x})$ is also not available and its intuitive interpretation (and hence the induction process) is depends on the *type of dataset*.

For discrete datasets

- In language modeling scenario, we interpret $\pi(\underline{x}|\overline{x})$ as a mechanism for *filling in* the extended context for a given short context.
- If $x =$ **chicago fire department**, then $\overline{x} =$ **department** and $\underline{x} =$ **chicago fire**
- Sampling from $\pi(\underline{x}|\overline{x})$ with $\overline{x} =$ **department** might also give us $x' =$ **uic ece department** or $x' =$ **uchicago cs department** in addition to **chicago fire department** of the original context.

Empirical Feature-Restricted Induced Models

a **core component** of the IMM methodology

- We then understand the empirical version of $\overline{Q}$ as doing a pass over the dataset, and summing the outputs of only those contexts that share the same short context as the original sample.
- We therefore have the following natural empirical version of $\overline{Q}$

$$\hat{Q}(y|\overline{x}) \propto \sum_t \mathbf{1}\{\overline{x}_t = \overline{x}\} Q(y|x_t)$$

- Instead of iterating over the entire dataset, we can precompute a dictionary

```
longFromShort['department'] = [
    ['chicago', 'fire'], ['uic', 'ece'], ['uchicago', 'cs']
]
```

Induced Model Matching

- In Logistic Regression, our regularization term was

$$\textbf{IMM-Risk}\Big(\underbrace{\overline{P}}_{\text{Bayes Optimal restricted model}}, \underbrace{\hat{Q}}_{\text{classifier's induced restricted model}}\Big)$$

- In Language Modeling, our regularization term will be

$$\textbf{IMM-Risk}\Big(\underbrace{\hat{P}}_{\text{Kneser-Ney bigram}}, \underbrace{\hat{Q}}_{\text{LSTM RNN's induced bigram}}\Big)$$

IMM in Language Modeling - Summary

Algorithm Sampled IMM with SGD for a Model Q with parameters W

Input: Tokenized data (x_t, y_t) for $t = 1, 2, ..., n$, $k =$ sampling rate
Output: IMM-trained model Q

repeat
  $Q(y_t|x_t) \leftarrow \textsc{FeedForward}(Q, \overline{x}_t, \underline{x}_t)$
  ∇_WCross-Entropy$(Q) \leftarrow \textsc{BackPropagate}(Q,$ Cross-Entropy$(Q))$
  $\hat{Q}(y_t|\overline{x}_t) \leftarrow 0$
  for all $x' \in$ **Sample**(extend$(\overline{x}_t), k)$ **do**
    $\hat{Q}(y_t|\overline{x}_t) \leftarrow \hat{Q}(y_t|\overline{x}_t) + \textsc{FeedForward}(Q, \overline{x}_t, x')$
  end for
  IMM$_t(Q) = -\sum_y \hat{P}(y|\overline{x}_t) \log \hat{Q}(y|\overline{x}_t)$
  ∇_WIMM$_t(Q) \leftarrow \textsc{BackPropagate}(Q,$ IMM$_t(Q))$
  $\textsc{ApplyGradients}(\nabla_W$Cross-Entropy$(Q) + \lambda \nabla_WIMM_t(Q))$
until convergence

Experimental results - LSTM RNN

measuring **feature-restricted** performance

Table: Restricted vs. Full Model on the Bigram Task (i.e., predicting the next word given *only* the previous one.)

Dataset	LSTM w/o IMM		LSTM w/ IMM	Kneser–Ney
Train	260.16	$\rightarrow$	237.55	92.19
Validation	339.24	$\rightarrow$	302.30	199.28
Test	309.56	$\rightarrow$	278.48	185.71

Experimental results - LSTM RNN

measuring **full-featured** performance

Table: Perplexity values on an LSTM RNN based Language Model. The numbers on None and KN Noising are from Xie et al and can be replicated using the original Stanford ML code. Like the baseline, for each row, we report the best value we could reproduce across many restarts.

Dataset / Model Size	Improvement	Validation	Test
PTB / 1500	None (only regular dropout)	81.6	77.5
	KN Noising (reproducible)	76.7	73.9
	IMM with KN Bigram	**76.0**	**73.3**

Experimental results - BERT

measuring **full-featured** performance

Table: Results on the $\text{BERT}_{\text{BASE}}$ Language Model. The baseline numbers can be replicated using the original BERT code by Google, as well as our provided repository. Matthew's Correlation Coefficient is used for CoLA and Accuracy for RTE. Like the baseline, reported numbers are averages across multiple restarts.

	$\text{BERT}_{\text{BASE}}$	+ MLM	+IMM
CoLA	52.1 ± 4.0	55.0 ± 3.0	$\mathbf{60.0 \pm 1.0}$
MRPC	88.9 ± 2.0	89 ± 1.0	$\mathbf{90 \pm 1.0}$
QNLI	90.5 ± 2.0	91.0 ± 2.0	$\mathbf{93.5 \pm 1.0}$
RTE	66 ± 3.0	68 ± 2.0	$\mathbf{71 \pm 1.0}$

Computational Tractability

Making IMM more efficient

ENGINEERING

Electrical and Computer Engineering

Sampling in the Main Loop

- Sampling k histories incurs a k-fold computational cost.
- In the thesis, we show that it's possible to do the **sampling implicitly in the main loop**, by decomposing the gradient:

$$-\nabla \sum_{\bar{x}} \pi(\bar{x}) \sum_{y} \overline{P}(y|\bar{x}) \log \overline{Q}(y|\bar{x})$$

$$= -\sum_{\bar{x}} \pi(\bar{x}) \sum_{y} \overline{P}(y|\bar{x}) \frac{\sum_{\underline{x}} \pi(\underline{x}|\bar{x}) \nabla Q(y|\underline{x},\bar{x})}{\overline{Q}(y|\bar{x})}$$

$$= -\sum_{\bar{x}} \sum_{\underline{x}} \pi(\bar{x}) \pi(\underline{x}|\bar{x}) \sum_{y} \overline{P}(y|\bar{x}) \frac{Q(y|\underline{x},\bar{x})}{\overline{Q}(y|\bar{x})} \nabla \log Q(y|\underline{x},\bar{x})$$

- **Constant-factor computational cost!** (with caveats, e.g., maintaining $\overline{Q}$)

Analysis

Connections with noising and reverse KD

UIC ENGINEERING

Electrical and Computer Engineering

Noising is single-sample IMM

- Recall the multiset (i.e. the "Python dictionary") from which we sample extended contexts, holding the short context (e.g. `department`) fixed.
- Since sampling is done uniformly, sampling a single different context is like sampling a different output
  - Think of it as an out of order pass over the data in the SGD
  - Sampling a different output is precisely what *noising* (Xie et al) does.
- **Single sample IMM is therefore noising!**

$$\sum_x \pi(x) \sum_y \overline{P}(y|\overline{x}) \log \frac{1}{\underbrace{Q(y|\overline{x}, \underline{x})}_{\text{key difference}}}$$

Reverse Knowledge Distillation is single-sample IMM

$$\sum_x \pi(x) \sum_y \overline{P}(y|\overline{x}) \log \frac{1}{\underbrace{Q(y|\overline{x}, \underline{x})}_{\text{key difference}}}$$

- Whereas noising does it implicitly, reverse KD explicitly adds this same objective to training a strong student (Q) with a weak teacher ($\overline{P}$) [4]
- Reverse KD does not exploit restriction being the nature of the weakness

Reference

[4] Yuan, L., Tay, F. E. H., Li, G., Wang, T., & Feng, J. (2020). Revisiting knowledge distillation via label smoothing regularization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3903–3911.

Noising and reverse-KD are upper bounds on IMM

Let's consider the **IMM Risk** and its upper bound

$$\underbrace{-\sum_{y}\hat{P}(y|\overline{x})\log\left(\mathbf{E}_X\left[Q(y|\overline{x}_t,X)\right]\right)}_{\text{IMM Risk}} \leq \underbrace{-\sum_{y}\hat{P}(y|\overline{x})\mathbf{E}_X\left[\log\left(Q(y|\overline{x}_t,X)\right)\right]}_{\text{Upper bound by Jensen's}}$$

- Computing $\mathbf{E}_X$ using a **single random sample**, we get a biased estimator of **IMM Risk** but an unbiased estimator of the upper bound (i.e. *noising* (Xie et al)).
- The *noising* objective is therefore an *upper bound* on the **IMM Risk**.
- Minimizing an upper bound introduces risk for suboptimality.

In the thesis, we show a numerical counterexample where the global minimizer $Q^\dagger$ obtained by *noising / reverse KD* is something other than P!

IMM in learning Markov Decision Processes

Incorporating POMDPs knowledge into MDPs

UIC ENGINEERING
Electrical and Computer Engineering

Markov Decision Processes (MDPs)

A quick review

Represented as a 5-tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$.

- State space $\mathcal{S}$: all possible states an agent can be in
- Action space $\mathcal{A}$: all actions an agent can take
- Either can be discrete or continuous

A **policy gradient** method will train the policy such that we maximize the reward when the agent traverses the environment.

- To do so, it will use T, the transition function, R, the Reward function and $\gamma \in [0, 1]$, the discount factor.

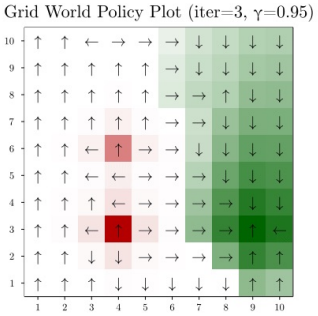


Figure: A simple grid MDP.

Partially Observable MDPs (POMDPs)

A quick review

- A Partially Observable MDP (POMDP) is a 7-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, O, \gamma \rangle$ with respective elements detailed on the right
- Additional elements are the observation space $\mathcal{O}$ and observation function O. The agent receives **observations** of the current state rather than the true state. Using past observations, it builds a **belief** about the underlying state.

Variable	Description
$\mathcal{S}$	State space
$\mathcal{A}$	Action space
$\mathcal{O}$	Observation space
T	Transition function
R	Reward function
O	Observation function
$\gamma \in [0, 1]$	Discount factor

POMDPs generalize MDPs

- POMDPs generalize MDPs
  - Any MDP is also a POMDP where all states are fully observable
  - Equivalently, the belief function of the corresponding POMDP collapses to a delta function.
- MDPs can be solved as POMDPs
  - A POMDP solver will simply treat an MDP like a POMDP
- Mixed Observability MDPs (MOMDPs), a special case of POMDPs
  - Have a mixed set of *observable* and *partially observable* states.
  - We use one toy example, however, our methodology applies to all POMDPs
  - We do not use the term Mixed Observability thereafter.

How POMDP policies are represented

- POMDP solutions, are called **conditional plans** (close to, but not the same as MDP "policies") and represented as sets of α vectors.
- One α vector per action, each of length equal to the number of states.
- The inner product of an α vector with the belief function gives us the **utility** of that action.

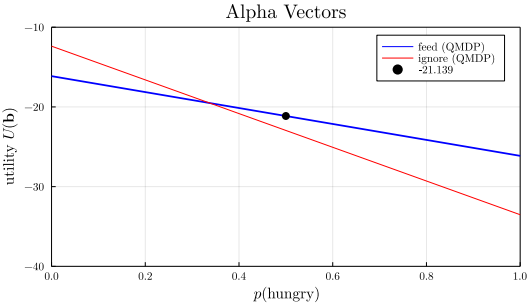


Figure: Solution of a "crying baby" POMDP which has just two states (hungry, fed) and just two actions (feed, ignore).

Toy example with POMDPs and MDPs

- A proof of concept, to transfer POMDP knowledge into an MDP.
  - Mask one dimension of a simple MDP (to get a POMDP)
  - Solve the POMDP using a POMDP solver (we use Fast Informed Bound [5]).
  - Do IMM-aided training of original MDP, incorporating POMDP solution into it.

Reference

[5] Kochenderfer, Mykel J., Tim A. Wheeler, and Kyle H. Wray. *Algorithms for decision making*. MIT press, 2022.

Toy example with POMDPs and MDPs

- The reward distribution is shaped like a mountain.
- The agent has to reach the center of the mountain peak.
  - MDP can look at both x and y coordinate.
  - POMDP policy looks only at x and maintains a belief over y
- POMDP policy obtained via FIB solver will be incorporated into MDP policy training (REINFORCE) using IMM

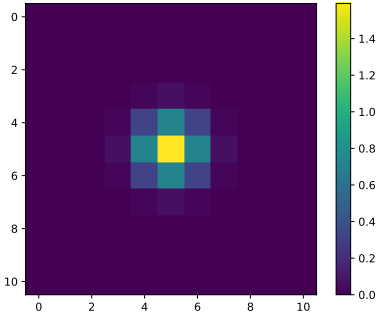


Figure: Reward distribution of "mountain climbing" MDP

Experimental Results with REINFORCE

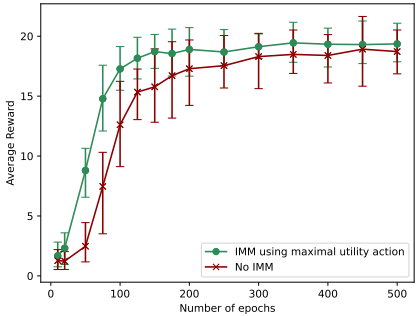


Figure: Performance of REINFORCE with and without IMM

What if we use a suboptimal target POMDP?

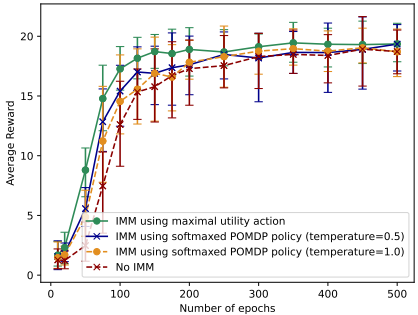


Figure: Performance of IMM with models of decaying quality

Toy example with POMDPs and MDPs

- Our ultimate goal is not to solve masked MDPs as POMDPs
  - Solving POMDPs is a well studied problem having a *curse of dimensionality* and a *curse of history* [5]
  - In practical applications a “POMDP solution” will be available to us.

Reference

[5] Kochenderfer, Mykel J., Tim A. Wheeler, and Kyle H. Wray. *Algorithms for decision making*. MIT press, 2022.

Where this can be practically useful

Merits of knowledge transfer from POMDP to MDP

- **Masked Datasets**: the lower dimensionality model was trained on (or constructed from) a dataset with some features masked due to privacy reasons
- **Continual/Lifelong Learning**: the lower dimensionality model was trained on (or constructed from) a dataset sampled from a previous policy having observations of smaller dimensionality
  - e.g., an old version of an autonomous vehicle with fewer sensors.

In either case, IMM gives us a method to transfer the knowledge of the restricted dimensionality model into the training of a larger one!

Summary

- This work contributes to a recent line of research that attempts to incorporate knowledge from weak models/teachers into the learning of stronger models/students.
  - We make connections between existing techniques in the domain, notably *data noising* and *reverse knowledge distillation*, that specifically try to use side-information from restricted-context models.
- We introduce a new approach, IMM, and show that using the restriction is crucial to have consistency in the limit (theoretically) and to achieve better performance with finite samples (experimentally).
  - While originally inspired by language modeling, we demonstrate the generally applicability of the IMM approach.

Publication

The thesis is based on one publication

Reference

Muneeb, U., & Ohannessian, M. I. (2024). Induced Model Matching: Restricted Models Help Train Full-Featured Models. *Advances in Neural Information Processing Systems*

Future Work

- IMM can potentially also aid **physics informed neural networks** in learning from structured models (physics equations).
- IMM can potentially be of interest to **increasing the context window of LLMs**, by letting shorter-window LLMs inform longer-window LLMs.
- IMM can have impact in more sophisticated RL examples and is of potential benefit in the domain of **curriculum learning** as well as **lifelong/continual learning**.

Thank you!

Proof of Inconsistency

Noising and Reverse KD can be asymptotically inconsistent

ENGINEERING

Electrical and Computer Engineering

Asymptotic Optimum $\neq P$

$$\underbrace{\mathsf{D}(P\|Q)}_{\text{Cross-Entropy}} + \underbrace{\lambda \sum_x \pi(x) \sum_y \overline{P}(y|\overline{x}) \log \frac{P(y|x)}{Q(y|x)}}_{\text{Single-Sample IMM}}.$$

- The single-sample IMM objective is no longer a proper KL divergence, even with P.
- This in itself doesn't mean the optimum cannot be P. However, we can come up with a counterexample where this risk becomes negative at a $Q^\dagger \neq P$.
- This means its minimum is $\neq P$. By convexity, it means the overall objective shifts away from P.