Exploring the Potential of Performance Bounds in Multi-Source Domain Adaptation

W.J.W. Bons

TU Delft

Exploring the Potential of Performance Bounds in Multi-Source Domain Adaptation

by

W.J.W. Bons

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday June 15, 2021 at 10:00 AM.

Student number:	4092716	
Project duration:	September, 2019 – June, 2021	
Thesis committee:	Dr. J. C. van Gemert,	TU Delft, PRB, chair
	Prof. Dr. M. Loog,	TU Delft, PRB, supervisor
	Dr. ir. R. C. Hendriks,	TU Delft, CAS
	Dr. J. A. Martinez,	TU Delft, CAS

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This work represents the result of my master thesis project at the Pattern Recognition and Bioinformatics group at the Delft University of Technology.

My years as a student have presented me with numerous interesting fields of study within science and engineering, and I am thankful for the chance to discover each of them. My studies have shaped me as an engineer; I have always desired to understand the *how* and *why*, but through ups and downs my studies have shown me how, though I appreciate a theoretical study, I value a practical focus. In light of this it is not surprising that I conclude my MSc programme in Signals & Systems with a thesis project in Machine Learning, which to me is a related field with a perfect practical focus.

After investigating many areas of Machine Learning, my main reason to focus on Multiple-Source Domain Adaptation (MSA) was to explain and extend current MSA literature. Explain, because I found that current literature dived head-first into theory. This work presents a thorough motivation for the MSA problem setting, i.e. under what assumptions it arises, and its practicality and relevance. I also aim to extend current literature, because the theory seemed oddly specific to one target model, the distribution-weighted combiner. This work clearly distinguishes between theory and assumptions that are inherent to the MSA setting and those required only for a specific target model.

The intention was to replace the distributions with other additional source knowledge and evaluate its effect in various scenarios. However, dissecting the theory proved to be non-trivial and the choice was made to focus on investigating what allows source knowledge to be used in new MSA theory. It is my hope that my perspective and explanations help readers who are new to this setting to more intuitively understand the MSA setting and theory, as well as inspire future extensions of this promising theory.

The perspective presented above is reflected in the document structure. The introduction will present a thorough motivation for the MSA problem setting and a full chapter is dedicated to investigating it. MSA theory is split into two chapters, to clearly separate the parts inherent to the setting and specific to additional knowledge.

This project has been a journey and I am thankful to have had wonderful people there to share it with. I am thankful for having had the opportunity to do real scientific research with a lot of freedom. For that I want to thank Prof. Dr. Marco Loog, as well as for his time and our enjoyable meetings. Furthermore, I want to thank the other members of my thesis committee, Dr. Jan van Gemert and Dr. Jorge Martinez, for their time in assessing my work. A big thanks goes out to the lovely people at PRB for the talks, coffee breaks and discussion–I have learned the most from you. I am especially grateful to those who kindly shared their office with me and made my time at PRB so much more than it could have been without them. During the covid-19 crisis I was lucky enough to study with Luuk and exchange our thesis struggles–thank you for helping me through. Finally, for support in many shapes and forms that long precedes this project, I want to thank my parents, Marjolein and Janne.

W.J.W. Bons
Delft, June 2021

Contents

1 Introduction

1.1. Motivation

1.1.1. Introduction to Research Question

Currently, machine learning models are prevalent and as a result trained models are readily available. The training data is also prevalent, but its sharing and hence its use can be discouraged for several reasons; firstly, because sharing data is complicated by privacy issues or bandwidth constraints (e.g. in embedded or distributed applications); secondly, because due to the large size of modern datasets it is too costly to store the data (e.g. for speech [1] or video data) or to train new models on it [2]. Therefore, it is increasingly relevant to research what is possible with already available trained models but without access to the training data.

An opportunity for Domain Adaptation presents itself when models are trained on *similar* datasets, i.e. having the same feature space but different distributions. This arises for example when for practical reasons samples are collected several times under different external circumstances. One example is medical data collected with the same equipment in multiple hospitals. Another example is in distributed applications where features correspond to physical location and the datasets are subsets of the same feature space. In both examples the datasets are part of a bigger machine learning problem and the models aim to solve the same task, i.e.: how to detect a disease from data collected with this medical equipment; and how to predict some property in the entire physical space, respectively. Their shared task and similar datasets imply it is perhaps possible to have a single model that can be applied on all datasets. This would certainly be useful and prompts the question: which of those models performs well on all datasets? In other words, it is a natural choice to set as target all of the sources.

This is a Domain Adaptation (DA) problem, the setting that intents to transfer knowledge from source domains to a target domain. In this work a *domain* refers to a distribution and an associated model trained on a realisation (dataset) of that distribution.

The goal is to construct a target model using only the trained source models and no source data. Either one of the sources can be selected as the target model, or the sources can be combined. Such combinations have been shown to improve performance over the combined models. [3][4] This work considers the weighted combiner that linearly combines the posterior class probabilities of the trained models h_k. For example, to rely more on model two than on model one, a combiner could be $h_T = 0.25h_1 + 0.75h_2$. Note that it includes as a special case selecting each individual model.

The issue with applying these (combined) source models on the target domain is that it violates the standard machine learning assumption that training and test data are similarly distributed. As a result of these different distributions source models do not generalise well to the target. [5] Here, as is common in Domain Adaptation, poor target performance is due to the DA algorithm used, so is called negative transfer.[6][7]

To approach this issue, many DA approaches use data to relate the source and target domains, for example by estimating a shift between the source and target distributions. In this work, by assumption no data is available to the DA algorithm. Therefore it is relevant to research the question:

> "How can pre-trained models be combined for performance on all their source domains, without access to data?" (RQ)

This is especially relevant when trained models are already available, because their application requires minimal effort; they need not be trained but can readily be applied.

The rest of this introduction is structured as follows. It will first be substantiated under which conditions the research question is relevant. This will lead to formulating the problem at hand as a Multiple-Source Domain Adaptation (MSA) problem setting. Following this, a performance bound from current literature is investigated, which, as this work will argue, has the potential to be extended from the perspective of 'additional knowledge'.

1.1.2. Assumptions of the MSA Setting

A reasonable assumption in practice is that the target is a mixture of sources. For example, if a new dataset consists of 80% data from dataset A and 20% data from dataset B, then its distribution is expected to be $\mathcal{D}(x, y) = 0.8\mathcal{D}_A(x, y) + 0.2\mathcal{D}_B(x, y)$. Note that this Target Mixture Assumption (TMA) includes each individual source as a target.

This work will discuss an *unknown* target, which is conveniently modelled by the TMA by assuming the weights to be unknown. The assumption that at test time it is unknown which source domain is the target is interesting for several reasons.

Firstly, it is a natural choice for the problem. Consider as an example that two research teams have collected medical imaging data of their own patients and trained a model to predict the presence of a disease. Both teams are trying to predict the disease from imaging data of *anyone*, not just their current patients; they want to generalise to new patients. The teams have the same task and it is desirable to have a single target model that can be applied on both source domains. It is then a natural requirement that a patient receives the same diagnosis in either hospital. The prediction should not rely on which source domain a patient is from. In some applications this is even impossible to know; what if, for example, a patient is found on the streets?

Secondly, it is practical to assume that at test time it is unknown which domain is the target. This allows deploying a single model rather than a set of models that the user still has to choose from.

Finally, it is an open question how to approach the research question without knowing the target. If the target is an unknown source, a model is available but can not be selected. If the target is a mixture of sources, no model exists trained on that domain. However, if it is known which source is the target, a model trained on that source is available and could be used as the target model.

In conclusion, the target is an *unknown mixture* of sources. Note that, contrary to most DA methods, no other target knowledge is assumed and no target data is required.

In this work the goal is to minimise a performance *guarantee*: an upper bound on the performance on any target. Optimising the performance on a single target instead is not possible because the target is unknown.

Minimising a performance guarantee addresses the issue of negative transfer, for the following reason. In DA settings, models can suffer from negative transfer. That is, models that intend to improve target performance can actually perform badly on the target. If the performance is guaranteed to be at most a certain value on *any* target, then by definition no negative transfer can yield a worse performance than this. Optimising for the best performance guarantee then minimises the worst possible negative transfer. In that way, this goal actively combats negative transfer. This is in contrast with the standard goal of minimising *expected* performance. Minimising the expected loss over all possible target distributions would allow negative transfer to be arbitrarily bad, regardless of good performance on average. Therefore the performance guarantee is minimised instead of the performance, even though this goal results in conservative solutions. (Additionally, optimising the expected loss is not possible because it requires knowing the distribution over all possible target mixtures.)

So, to combat the important and expected issue of negative transfer, the goal is to find a target model that provides a good performance guarantee, i.e. a small expected loss that can be guaranteed for all possible target mixtures. This can be interpreted as a robustness property of the model, as it is robust against any target mixture that it might be applied on.

To summarise, this work assumes that (a) no data is available, only trained models, (b) the target is an unknown mixture of sources, and (c) the goal is to minimise the performance guarantee. This describes a Multi-Source Domain Adaptation (MSA) setting, [1, 8, 9] which will therefore be used to model the problem.

1.2. Existing Method and Contributions

Existing theory for the MSA setting gives a performance guarantee for a combiner whose weights depend on (estimates of) the sources' joint distributions, and this performance guarantee is better than for a combiner that does not use the distributions. [1, 8, 9] By using the joint distributions, the combiner accounts for the difference in train and test distributions. However, the distributions might not be the only mismatch between trained models and what domain they are tested on in the MSA setting; anything related to the training procedure of source models influences the target model's performance and so might be used beneficially in the combining weights. For example, the complexity of the source models also influences the target model's performance and is not considered by current MSA theory. Thus, in addition to the in the MSA setting available source models, some *additional knowledge* of the source domains might be used in the combining weights. This work presents MSA theory from the perspective of additional knowledge, in general, and hypothesises that other knowledge than the distributions can be useful as well, thereby expanding MSA theory.

Any additional knowledge is related to the source domains, not to the target. Using such knowledge to combine suits the MSA setting, because it does not require target knowledge or the sharing of source data.

The scientific contributions of this work are threefold. Firstly, to explain MSA theory from the perspective of additional knowledge, this work explores and explains the MSA setting and the behaviour of models in it, as well as current MSA theory and its performance bound. In current literature, this bound is simply proven to hold and used, and a second contribution is to clarify its assumptions and in particular to make a clear distinction between theory and assumptions that are inherent to the MSA setting and those that follow from the current choice of distributions as additional knowledge. This work will not propose combiners that use additional knowledge, but rather it will investigate what makes source knowledge beneficial in the MSA setting and of use in new MSA theory. With this final contribution this work aims to inspire future research on similar performance bounds with other types of additional knowledge.

1.3. Research Questions

The main research question is:

> "How can pre-trained models be combined for performance on all their source domains, without access to data?" (RQ)

In the previous section it was motivated that this question can be modelled as an MSA setting. (The performance on an unknown mixture target is of interest, which is modelled as a performance guarantee for all possible targets.) The sub-questions are formulated as follows:

> "Which combinations of pre-trained models are robust in the MSA setting?" (RQ1)

It will be shown that some target models are not robust to all possible targets and result in negative transfer. Therefore this is followed by:

> "How can the existence of a robust model be determined to counteract negative transfer in the MSA setting?" (RQ2)

> "What could be used as additional knowledge to ensure satisfactory robustness of a model in the MSA setting?" (RQ3)

That is: (RQ2) investigates current theory and (RQ3) works towards expanding it.

1.4. Scope

In this work the research question will be considered within the following scope.

The learning task is supervised classification.

Data has continuous features $\mathcal{X} \in \mathbb{R}$ and discrete labels $y \in \{1, 2\}$ that are the same for all domains.

Models (or 'hypotheses') output a posterior class probability for each class (sometimes called 'soft labels'). Because two classes are assumed and posterior probabilities sum to one, in practice only one probability is sufficient as an output, not a vector. Thus, models are defined as $h_k : \mathcal{X} \times \mathcal{Y} \to [0, 1]$.

The loss function is the log-loss (also known as cross-entropy loss) in all domains.

The sample size of target domains is outside of the scope of this work and in experiments will be taken large. That is, effects of sampling a specific target dataset are not investigated, only the expected (mean) performance over all datasets for each possible mixture parameter. One way to view this is that the goal is (expected) performance on a target mixture *distribution*, not on a realisation of it.

When using distributions as additional knowledge, the true distributions are used. (The influence of density estimation on MSA theory was formalised by [8].) Availability of additional knowledge is assumed the same for all sources; distinction is out of scope.

It should be noted that the MSA setting and theory are applicable to a broader extent. For example, [1] considers regression models and [9]'s proofs do not assume that the models output the posterior distribution, simply a distribution. However, this broader perspective is not the point of this work. To investigate the setting and additional knowledge it was chosen to focus on the scope outlined above.

1.5. Related Work

Multiple-Source Domain Adaptation

In Multiple-Source Domain Adaptation settings, a model trained on source domains is to be adapted for good performance on target domains. Typically, unlike in the MSA setting, labelled source data and unlabelled target data are available during training. [10] This allows relating the targets to the sources and adapting accordingly, for example by estimating a domain shift or by learning a domain-invariant representation. [11] Having the source data also allows naively merging it and training a single predictor on the merged data. In contrast, in the MSA setting no data is available, only trained models, and the targets are unknown but constrained with respect to the sources. [1, 8, 9]

(For a brief overview of related problem settings, see section 5.2.)

MSA Setting

In the MSA setting a sufficiently robust distribution-weighted (DW) combiner always exists, guaranteeing an expected loss of at most ϵ, the worst-case expected loss of a source domain. [9] This setting was first formulated and its theory analysed and proven under different assumptions, [9] i.e. in a deterministic setting where hypotheses are target functions instead of a probabilistic setting where they output posteriors, and for other loss functions than the log-loss. [1, 8] In that deterministic setting, even if the target is known, the combining weights should depend on the marginal source distributions since (a) there exists a problem for which any combiner with fixed weights performs poorly and (b) using a distribution-weighted combiner guarantees an expected loss of at most ϵ.

This upper bound also holds when the target is unknown [1] and similar bounds hold if the following assumptions are relaxed, [8, 9] lending more practical use to the theory. Firstly, relaxing the target distribution assumption from a mixture of sources to any unknown target distribution yields bounds that include the Rényi divergence between the target distribution and the closest mixture of sources. Secondly, if the assumption that the true source distributions are known is relaxed to having approximations of them, a bound utilises the Rényi divergence to measure their closeness to the true distributions. This shows the effect of sample size in the MSA setting.

For a probabilistic setting, theory similar to that for the deterministic setting exists. [9] The distribution-weighted combiner then depends not on the marginal but on the joint source distributions.

Theory for the cross-entropy loss gives similar bounds to other loss functions [9, apx.C] although their derivation differs slightly. Firstly, some earlier proofs rely on a triangle inequality [8] that is not satisfied by the cross-entropy loss. [12]. Furthermore, in the probabilistic setting the DW-combiner is slightly adapted to ensure that it is normalised, [9] i.e. that it outputs a valid probability vector, as required for the cross-entropy loss. [3] A slightly altered derivation still yields the same bounds. [9]

An algorithm exists that finds the weights of the DW-combiner which satisfies the theoretical bound. [9]

Negative Transfer

In Domain Adaptation, negative transfer is the phenomenon of deteriorating target performance when trying to transfer from a source with the intention of improving target performance. [7, 6] Negative transfer is the result of the chosen algorithm and is caused by differences in joint distributions. [6]

Classifier Combinations

A combination of classifiers (combiner, ensemble model) can improve performance over the combined models. [3] Combining pre-trained models can aid performance in multiple-source DA, [4] which is not explained by ensemble effects, [5] but rather by DA effects such as capturing domain-specific bias.

1.6. Document Outline

This introduction has presented a motivation for several assumptions and a goal, which has led to formulating the research question in the MSA problem setting. First, chapter 2 will formally define this setting and investigate the behaviour of models in it. Chapter 3 will then introduce a performance bound of MSA theory, after which chapter 4 will show how this theory can be used when assuming a combiner that uses additional knowledge.

The problem setup is first explored in detail to develop intuition for it (RQ1) and to allow interpretation of MSA theory (RQ2). Following this, additional knowledge in MSA theory is discussed (RQ3).

2

MSA Setting and Combining

2.1. Introduction

Chapter 1 has presented a motivation for treating the main research question

> "How can pre-trained models be combined for performance on all their source domains, without access to data?" (RQ)

as a problem in the MSA setting. This chapter will investigate this setting by asking the question:

> "Which combinations of pre-trained models are robust in the MSA setting?" (RQ1)

To answer this, this chapter will first discuss the MSA setting, starting by defining its assumptions and goal. Next, by example the robustness of source models and their combinations is evaluated and compared. This investigates whether robustness can and does improve by using a combination instead of the source models. Along the way, these examples clarify the behaviour of models and the to-be-optimised goal in this setting. This leads to the conclusion that non-robustness is a disadvantage of the setting for source models as well as some combiners, but also that robust models are possible.

2.2. Assumptions and Notation

Let the input space $\mathcal{X} \subseteq \mathbb{R}$ consist of continuous features and the output space $\mathcal{Y} \subseteq \{1, 2\}$ of binary labels.

A domain D refers to a distribution $\mathcal{D}(x, y)$ and an associated model h that was trained on a realisation of $\mathcal{D}(x, y)$. No data of domains is available to train on or test with, only the trained model is.

The domains D_k, $k = \{1, ..., K\}$ are called *source domains* and the *target domain* D_T is assumed to be a *mixture* of them, as defined by the *Target Mixture Assumption* (TMA). The TMA is assumed linear and mixes the joint distributions, i.e.

$$\mathcal{D}_T(x, y; \lambda) = \sum_k \lambda_k \mathcal{D}_k(x, y) \tag{2.1}$$

with true mixture weights $\lambda \in \Delta = \{\lambda \in \mathbb{R}^K : \lambda_k \geq 0 \wedge \sum_{k=1}^K \lambda_k = 1\}$ (which implies $\lambda_k \leq 1$ or $\lambda_k \in [0, 1]$). Mixture weights other than the true (i.e. the true target in a scenario) are denoted by some variable other than λ, e.g. z. Note that each individual source $\mathcal{D}_k$ is also an 'extreme' mixture (with exactly one of the mixture weights non-zero and equal to one). (For ease of reading we also use the notation $\mathcal{D}_\lambda := \mathcal{D}(x, y; \lambda)$ and also abuse some notation by using integer k as a mixture parameter, i.e. $\mathcal{D}_k$ refers to either the source or equivalently to that source viewed as an extreme mixture.)

Models (or 'hypotheses') $h : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ have as output for each $x \in \mathcal{X}$ the vector of posterior class probabilities $p(y|x)$. A model parameterised by a parameter z is denoted $h(z) = h(x, y; z)$.

The loss of a model's prediction is measured by the loss function $\ell : [0, 1] \to \mathbb{R}_+$. It is non-negative and as a result any risk (expected loss) is non-negative $L \geq 0$. Specifically, the log-loss[13] is used: $\ell(x, y) = -\log \hat{p}(y_{\text{true}}|x)$ where $\hat{p}(y_{\text{true}}|x)$ is one element of $|\mathcal{Y}|$-length vector $h(x)$.

Hypothesis Space

Recall that hypotheses h output posterior probabilities, i.e. $h(x, y)$ is for any $x \in \mathcal{X}$ a vector $\hat{p}(y|x)$ of length two. In this chapter[1] the hypothesis space is that of linear combiners, i.e. for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$

$$h_T(x, y) = \sum_k z_k h_k(x, y) \tag{2.2}$$

with weights $z_k \in \mathbb{R}$ that are constant (i.e. do not depend on e.g. x or y or some parameter). The weights $z_k \geq 0$ are assumed normalised $\sum_k z_k = 1$ which implies $z \in \Delta$, i.e. they are valid mixture weights. Equation 2.2 is also denoted as the weighted combiner $h_W(z)$.

2.3. Property of Losses

From the setting defined in section 2.2 follows a behaviour of models in this setting, as stated now. (Note that this holds for *any* model h, not just for combiners.) It shows that the linear relationship between source and target distributions assumed by the TMA carries over to the expected losses. This relationship will be used in MSA theory (ch. 3). It is also a useful property of the MSA setting in practice, since it allows calculating the loss on any target from the source losses, without needing target or source data.

Lemma 1. *The target loss is*

$$L_{\mathcal{D}_\lambda}(h) = \sum_k \lambda_k L_{\mathcal{D}_k}(h)$$

for ℓ the log-loss and for any h and λ_k.

Proof.

$$L_{\mathcal{D}_\lambda}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}_\lambda}\{\ell\left(h(x, y)\right)\} = \sum_y \int_{-\infty}^{\infty} \mathcal{D}_\lambda(x, y)\ell\left(h(x, y)\right)\, dx$$

for discrete y and continuous x. Substituting the TMA yields:

$$= \sum_y \int_{-\infty}^{\infty} \sum_k \lambda_k \mathcal{D}_k(x, y)\ell\left(h(x, y)\right)\, dx$$

where for well-behaved h the integral and sum over k can be interchanged:

$$= \sum_k \lambda_k \sum_y \int_{-\infty}^{\infty} \mathcal{D}_k(x, y)\ell\left(h(x, y)\right)\, dx = \sum_k \lambda_k L_{\mathcal{D}_k}(h)$$

□

2.4. Formal Goal

In the MSA setting considered, the goal is to perform well on all source domains as a target. This is modelled (as motivated in chapter 1) by the Target Mixture Assumption, i.e. the target domain D_λ is a mixture of the source domains D_k, and this mixture is unknown.

Performance is measured by expected loss over $\mathcal{X} \times \mathcal{Y}$, as is standard in machine learning. In this setting, the objective is to minimise expected loss on the *target* domain, i.e. to minimise $L_{\mathcal{D}_\lambda}(h_T)$. The model h_T does not depend on k or λ, because the same model is desired for any target domain, respectively λ is unknown.

In this work, λ is unknown and so, as motivated in chapter 1, the goal is to provide not good performance but a good performance *guarantee*. Let a performance guarantee of a model h be defined as an upper bound on its performance on *any* target mixture D_λ:

$$\forall \lambda \in \Delta: \quad L_{\mathcal{D}_\lambda}(h) \leq L_{\mathcal{D}_{\lambda_{worst}(h)}}(h) \tag{2.3}$$

[1] Chapter 4 will extend this to linear combiners whose weights depend on x, y.

where

$$\lambda_{worst}(h) = \arg\max_{\lambda} L_{\mathcal{D}_\lambda}(h) \tag{2.4}$$

which is given by the loss on some mixture on which the model performs worst: a worst-case mixture *for the model*. In other words, h is guaranteed to achieve at least this performance regardless of which target it is applied on. An interpretation is as a robustness property of h, as that model is robust against any target mixture it might be applied on; again in the sense that its performance is guaranteed at least as good as the guarantee. The terms 'performance guarantee' and 'model robustness' will be used interchangeably in the rest of this work.

This work has as goal to find a target model h_T that provides a good performance guarantee, as in equation 2.3. The best performance guarantee is that with the lowest expected loss, so the objective is:

$$\boxed{\min_{h_T} L_{\mathcal{D}_{\lambda_{worst}(h_T)}}(h_T)} \tag{2.5}$$

where $\lambda_{worst}(h_T)$ is defined by equation 2.4. Note well that the worst-case mixture depends on the model used so is not the same for the entire hypothesis space.

In conclusion, the setting's goal is optimised for by the optimisation problem in equation 2.5, which finds the target model that is most robust to application on any target mixture (or: the model with the best performance guarantee).

Related

Alternative MSA settings (outside of the scope of this work) are possible if some knowledge of λ is available. If, for example, λ was known, it could simply be used to optimise the expected loss:

$$\min_{h_T} L_{\mathcal{D}_\lambda}(h_T)$$

If not a value but a distribution over λ was known, a Bayesian approach would be to minimise its expected value:

$$\min_{h_T} \mathbb{E}_{\lambda \sim p(\lambda)} L_{\mathcal{D}_\lambda}(h_T)$$

Another alternative is a minimum regret-based approach.

Hypothesis Space

To find the most robust target model (eqn. 2.5) a hypothesis space must be chosen to search in. As motivated in chapter 1, a combination of the source models is used. That is, a parametric form of the combiner is defined and thereby a family of models is defined as hypothesis space from which the optimal is to be found. This hypothesis space includes the set of trained source models h_k, so a possible solution is to select one of them as h_T.

In this chapter a linear combiner is used, as defined by equation 2.2.

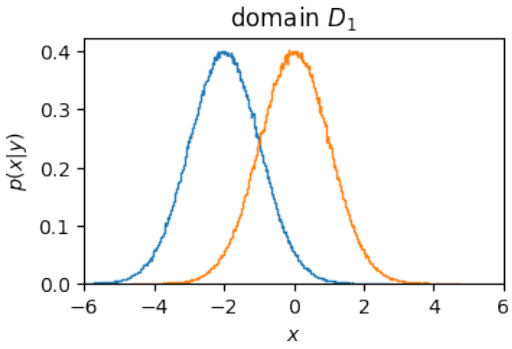
2.5. Discussion by Example

2.5.1. Experimental Setup and Baselines

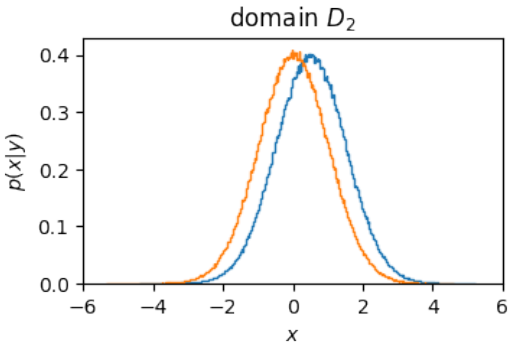
To develop intuition for the setting simple domains are used as an example. They consist of two gaussian classes in a one-dimensional feature space, where one class has its mean shifted between domains. Example realisations are shown in figure 2.1.

Unless otherwise stated, all performances reported below are on an independent test set and sample sizes are taken large enough to approximate the true distribution.

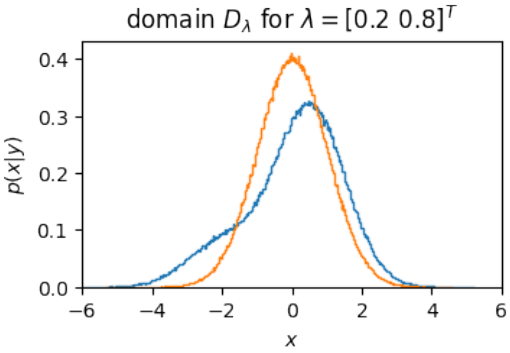
Target domains are generated as follows: after splitting each source domain in a training and test set (50% each), the training sets are mixed together by selecting a subset of the source training sets so that the resulting mixture follows the TMA with the correct λ, and also has the same sample size as the source domains. The same procedure produces the target's test set.



(a) Domain D_1 with $p(x|y_1) \sim \mathcal{N}(-2, 1)$ and $p(x|y_2) \sim \mathcal{N}(0, 1)$.



(b) Domain D_2 with $p(x|y_1) \sim \mathcal{N}(0.5, 1)$ and $p(x|y_2) \sim \mathcal{N}(0, 1)$.



(c) Domain D_λ for $\lambda = \begin{bmatrix}0.2 & 0.8\end{bmatrix}^T$.

Figure 2.1: Example realisations ($N = 2 \cdot 10^6$ per domain) of two source domains and one possible target mixture. Binary classification in 1D (one feature). Each class has a gaussian distribution (parameters below the relevant figures) and all class priors and variances are equal. The source domains differ in the mean of one class, which is shifted asymmetrically.

Baselines

To investigate the behaviour of models in the MSA setting, they are compared with the following baselines.

If the target is one of the sources, that source's own model is a baseline. After all, the target model h_T is to be used in that domain in place of the source model h_k that is already available. It is desired that the new target model is at least as robust as the source model.

Another baseline are the other sources' models. After all, since they are given in the setting, the easiest construction of a target model is to select one of the source models. This was also part of the motivation for the research question and it was hypothesised that some source models can also be robust to other targets.

All together comparison with all source models is in order. Especially the source model that provides the best guarantee is of interest, because of the easiest target models it is the one that best achieves the setting's goal.

2.5.2. Source Models and the Best Guarantee as the Setting's Goal

First, consider selecting either available source model h_k as the target model to use. Figure 2.2 shows the performance of linear source classifiers on all possible mixture targets D_λ. This shows (for this example) the expected behaviour when models are applied on datasets with different distributions than they are trained on. That is, the source models perform the best on their own domains (e.g. h_1 for $\lambda_1 = 1$) and worse on mixtures that include less of their own domain. Consequently, if for example h_1 instead of h_2 is applied on D_2, the expected loss increases from 0.66 to almost 1.8.

Now consider the goal: to select the model that provides the best performance guarantee for all

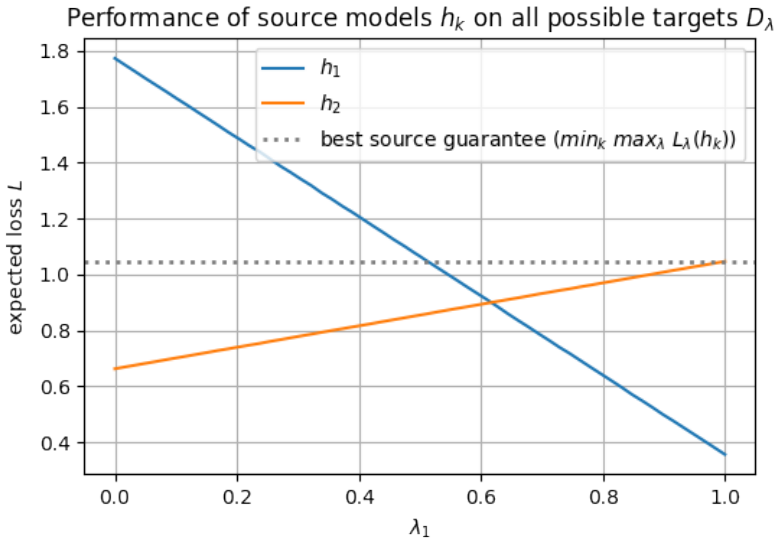


Figure 2.2: Comparison of the performance of linear source classifiers h_k on all possible mixtures targets D_λ. The goal of the setting is to provide the best guarantee, which is here provided by h_2. The domains used are shown in figure 2.1, the sample size of domains is $N = 2.5 \cdot 10^6$ and the gridsize for λ used was 60.

possible targets, i.e. the lowest loss for the worst λ(eqn. 2.5). h_1 and h_2 provide guarantees of respectively almost 1.8 and 1.05. Therefore h_2 is preferred over h_1, even though h_1 would have performed better on target mixtures with $\lambda_1 > 0.61$. By selecting $h_T = h_2$, the (expected) loss on any target can be guaranteed to be at most $L_{D_\lambda}(h_T) \leq 1.05$.

2.5.3. Combiners as Compared to Source Models

Consider now a weighted combiner (eqn. 2.2) in this setting. Figure 2.3 shows all possible combiners, i.e. all weights z, and compares their performance as well as performance guarantees with the source models and relevant baselines.

Firstly, consider the combiner's performance on a single target D_λ (fig. 2.3a). For these example domains, on this target some combiners perform better than the best source model (h_1), while others perform worse, upto at worst the performance of the other source model (h_2). So for this target, choosing the right combiner (for example the optimal) outperforms both source models.

Secondly, consider the combiner's performance guarantee, i.e. its performance on the worst-case target (fig. 2.3b). Similar to its performance on a single target (fig. 2.3a) the performance guarantee of some combiners is better than the best guarantee of the source models, while other combiners have a worse guarantee. This means that the weighted combiner $h_W(z)$ is not necessarily better than the source models in this setting: for the 'wrong' weights z, a source model will provide a better guarantee. For other weights z, however, the combiner provides a better guarantee than the source models; in figure 2.3b for any $z < 0.82$. (Chapter 3 will study how to find those robust models.)

Effect of Source Model on Combiner: Complexity and Data

Logically, the source models greatly influence the combined model. When using a combiner instead of a source model, the following differences between them should be considered.

Firstly, the combiner is often but not always more complex. For example, consider as source models two decision trees that makes five decisions (different for each source domain). A linear weighted combiner of the posterior probabilities then makes ten decisions and thus has a more complex decision boundary. Consider now that the decisions made are the same for both sources; then the combiner only makes five decisions (albeit weighted differently), the same number as the source domains. Lastly, in some cases the combiner can also be less complex. For example, if the source models make opposing decisions then a uniformly weighted linear combiner makes no decisions at all.

In this work some examples use linear source classifiers, which, because their posteriors are combined linearly, results in a target model that is also linear. Any effects on for example performance are then not due to an increase in complexity.

Note that even though the model can become more complex by combining, the problem to solve, i.e. performance on a $\mathcal{D}(x, y)$, becomes more complex as well. This is in contrast with the common approach of using combiners to improve performance on a single dataset; there only the model's complexity increases, the problem's is unchanged, and performance often improves as a result. It is therefore not trivial that in the MSA setting the combiner (with possibly the same complexity as the source models) is able to solve the more complex target problem. Yet, such a result will be achieved in section 4.2.

Secondly, as compared to the source model, the combiner is (indirectly) trained on more training data, possibly with different qualities such as it being from a different part of the feature space. That can be a partial explanation for any effects and performance differences between combiner and source models.

2.5.4. Negative Transfer

For both selecting a source model as target model as well as for a combiner, it was empirically shown that performance can suffer greatly in the MSA setting, as compared to using a source's own model. This is negative transfer, because the deteriorating performance is the result of the chosen MSA algorithm[6] that produces the target model, and because performance of the algorithm that intended to improve performance on other domains has actually worsened that performance.[7]

Because the performance guarantee is by definition the performance on the worst target (eqn. 2.3), a bad performance for *some* target due to negative transfer results in a bad performance guarantee. In other words, as a result of performance decrease (which is negative transfer) the performance guarantee also suffers and the non-robustness of the target model is viewed as negative transfer.

2.5.5. Optimal Performance Guarantee

For the weighted combiner as example model, figure 2.3b also shows how the performance guarantee and its optimum behave.

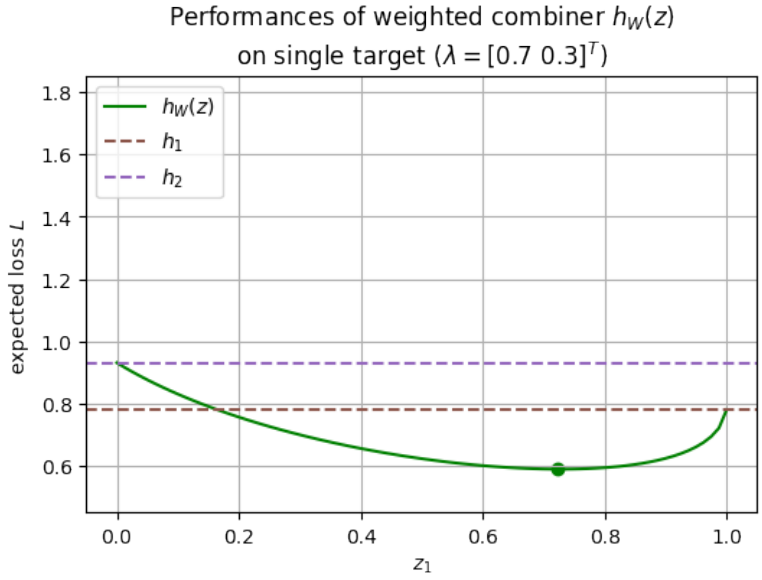
For reference the performance on two single targets (as well as the range for all possible targets) is plotted. Firstly, this shows how the guarantee is constructed as a point-wise maximum over all λ. Secondly, for all λ the guarantee passes through the optimum of the worst-case lambda. This implies that if the optimal z is chosen, then the performance on *all* possible targets is the same and that performance equals the performance guarantee. On the contrary, if a non-optimal value of z is chosen, then the performance on the possible targets varies: the loss on some targets will be larger and on some smaller than the optimal guarantee's loss. One interpretation of this is that to minimise the guarantee some good performance on 'easy' mixtures is sacrificed to improve bad performance on 'hard' mixtures, until performance on all mixtures is equal in the optimum.

2.6. Conclusion

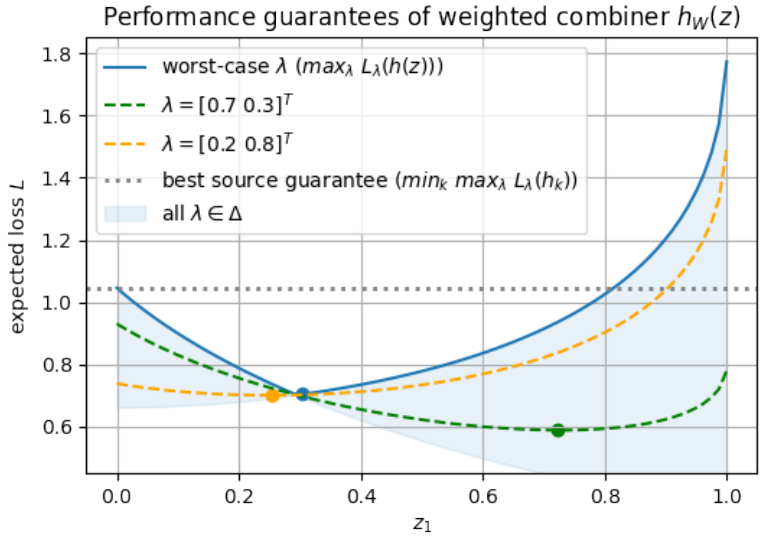
This chapter has investigated which combinations of pre-trained models are robust in the MSA setting. The setting was first defined formally. Notably, inherent assumptions of the setting were noted to be the availability of trained models and no data, a target that is a mixture of sources, and as goal to find a robust target model, i.e. that minimises the worst expected loss for any unknown target. The target model is a linear combination of the source models' posterior probabilities. It was further proven that that target loss is a linear combination of the source losses and by example it was shown that for optimal z the weighted combiner achieves the same loss on all targets.

It was then empirically shown that the source models are less robust on other domains than their own, and that a weighted combiner can be either less or more robust than the source models. Worse performance of the target model than the source models is negative transfer in the MSA setting. This translates to non-robustness of the target model, which is therefore negative transfer and is a disadvantage of the MSA setting. However, the same example has shown that a weighted combiner can also be *more* robust, i.e. provide a better performance guarantee, than the source models.

The question then becomes how those robust models can be found, which is the topic of chapter 3.



(a) Single target with $\lambda = [0.7\ 0.3]^T$.



(b) Worst-case target $\lambda = \arg\max_\lambda\ L_\lambda(h(z))$ i.e. performance guarantee. For comparison performances (not guarantees) on two targets as well as the range for all targets are shown.

Figure 2.3: Comparison of all weighted combiners $h_W(z)$ (equation 2.2 for all z) with linear source classifiers h_k when used on either a single target (fig. 2.3a) or the worst-case target (fig. 2.3b). The performance on the worst-case target is a performance guarantee and minimising it is the goal of the setting. The domains used are shown in figure 2.1, the sample size of domains is $N = 2 \cdot 10^6$ and the gridsize for z and λ used was 80.

3

MSA Theory Guarantees Robustness

3.1. Introduction

In the previous chapter the MSA setting was introduced, notably its assumptions and goal: to find the target model with the best performance guarantee, i.e. that is most robust to any target mixture. Some examples have shown that the source models are less robust on other domains than their own, and that some weighted combiners (some combining weights) are less robust than the source models. Other combiners were seen to be more robust than the source models. This showcased negative transfer as a disadvantage of the setting, but also that some combiners showed promise to counteract this negative transfer. This motivates the question:

> "How can the existence of a robust model be determined to counteract negative transfer in the MSA setting?" (RQ2)

First, this chapter will derive a theorem from literature (sec. 3.2) that guarantees a robust model in the MSA setting. This is done for a more general model–it is not even assumed to be a combiner–so the theorem is fundamental to the setting. After its derivation a discussion follows on using the theorem in the MSA setting (sec. 3.3), specifically on its relation to the setting's goal and on negative transfer.

3.2. MSA Theory

It is now proven that a robust model $h(p)$ is guaranteed to exist. Specifically, regardless of h, if h is continuously parameterised by a mixture parameter p then a parameter value of p exists for which the performance on any mixture is bounded. (Discussion on this bound follows after its derivation.)

Assumptions

For any $h(p)$ parameterised by a mixture parameter p it is assumed that $L_{\mathcal{D}}(h(p))\ \forall \mathcal{D}, h$ is continuous in that p.

The loss function $\ell\left(h(x,y)\right))$ is assumed convex.

Lemma 2. [1] [2] *Consider K sources $\mathcal{D}_k$. There exists a $p \in \Delta, p_k \neq 0$ such that for any source k*

$$L_{\mathcal{D}_k}(h(p)) \leq \sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta' \tag{3.1}$$

for a model $h(p)$ that is continuous in its parameter $p \in \Delta$ and for any $\eta' > 0$.

Proof. See page 33.

[1] For $h(p)$ the DW-combiner $h_{DW}(z)$ of equation 4.2 this lemma and its proof are mostly equivalent to [9, sec. B.3, lem. 6] which in turn is based on theorem [1, lem. 2]. Specific equivalences or differences are mentioned when they're relevant.

[2] A more general version of this lemma (for other fixed-point mappings) is given in appendix C as lemma 8.

This lemma is a general statement about losses on the *source* domains. By now applying the TMA a similar statement is obtained about losses on two *mixtures*.

Theorem 3. *Consider two mixtures $\mathcal{D}_p$ and $\mathcal{D}_q$. There exists a $p \in \Delta$, $p_k \neq 0$ such that for any $q \in \Delta$*

$$L_{\mathcal{D}_q}(h(p)) \leq L_{\mathcal{D}_p}(h(p)) + \eta' \tag{3.2}$$

for a model $h(p)$ that is continuous in its parameter $p \in \Delta$ and for any $\eta' > 0$.

Proof. See page 34.

Lemma 4. *In theorem 3, the tightness of the bound in equation 3.2 (i.e. the difference between its left- and right-hand side) is bounded by:*

$$\frac{\eta'}{K} < L_{\mathcal{D}_p}(h(p)) + \eta' - L_{\mathcal{D}_q}(h(p)) \leq \frac{\eta'}{K} \sum_{k=1}^{K} \frac{1}{p_k} \tag{3.3}$$

Proof. See page 34.

Several remarks are in order about these statements. This focuses on theorem 3 but is equally valid for lemma 2: note that equation 3.1 is obtained from equation 3.2 by taking $q = k$ and applying the TMA. This focus on mixtures over sources is chosen because the setting's goal is to optimise for mixtures and because they are more general than sources.

Existence

To begin with, note that the bound only holds for (at least) *one* value of p and *not* for any value of p. That is, for a single model $h(p)$, *not* for the entire family of models $h(p)$. The existence of that model $h(p)$ is guaranteed by theorem 3.

Robustness

Secondly, the theorem's bound (eqn. 3.2) is a robustness property of the model $h(p)$: it bounds the loss of $h(p)$ on any mixture. To find a model with such a robustness property is the goal of the setting. (To be precise, equation 3.2 is a statement as equation 2.3.) However, the theorem does not say *how* robust it is; the value of the upper bound $L_{\mathcal{D}_p}(h(p))$ is unknown. This value might be very large,[1] resulting in a loose, vapid bound. Section 3.3 will explore both how the bound relates to the setting's goal as well as how the upper bound can be quantified, and furthermore how tight the bound is (cor. 4).

Scope

These results are fundamental to the MSA setting considered here. Firstly, no assumptions were made about the models other than continuity in the mixture parameter. For example, nothing was assumed about model combinations. Secondly, for the proofs a crucial assumption was the TMA, which was used both indirectly in the fixed-point mapping of lemma 2 and directly in theorem 3. Thus another setting with a different TMA would require a different fixed-point mapping to derive a similar result. (See appendix C for a discussion on this.)

3.3. Discussion

Recall that the theorem's bound (eqn. 3.2) is a statement as equation 2.3, i.e. a robustness property of a model such as finding one is the goal of the setting. Further recall that having such a robustness property counteracts negative transfer in the setting (sec. 1.1.2) by being ensured that performance is never worse than that guarantee. Thus the theorem can be used in the setting to counteract negative transfer.

The theorem guarantees that a model exists that has as performance guarantee $L_{\mathcal{D}_p}(h(p)) + \eta'$. Let us discuss of this guarantee firstly its tightness and secondly its value, and furthermore discuss how to find the model with this property.

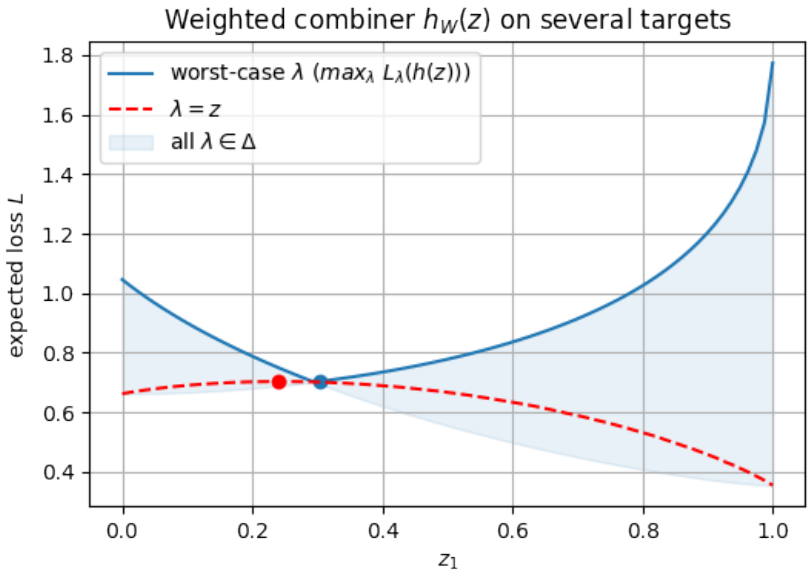


Figure 3.1: Comparison of setting's goal and theory's upper bound for the weighted combiner (eqn. 2.2). To minimise the loss for the worst-case λ is the goal of the setting; Theorem 3 guarantees that for some z the line $\lambda = z$ upperbounds the loss on the worst-case λ, which in this example is achieved for $z = 0.30$. (The same example domains and experimental settings were used as in figure 2.3.)

3.3.1. Tightness of the Bound

The tightness of equation 3.2 is bounded by corollary 4. This shows that for some p, q and sources $\mathcal{D}_k$ the bound is almost tight: there is a gap of η'/K where $\eta' > 0$ can be chosen arbitrarily small. So the robustness property in theorem 3 is almost (for small η') the tightest possible under the current assumptions, i.e. to derive a robustness property that holds for any sources $\mathcal{D}_k$ and any continuous hypothesis h.

Furthermore, recall that from a previous example (sec. 2.5.5, fig. 2.3b) it was hypothesised that a target model's loss is the same on all targets. Corollary 4 sheds some light on this hypothesis, which is also claimed in passing by [1, sec.5.2][9, sec.4.1]. By choosing η' to be small, by the corollary the loss of $h(p)$ is approximately the same on all mixtures $\mathcal{D}_q$, except possibly if the guaranteed p has a p_k close to zero. This exceptional case requires further analysis to prove the same loss on all targets, or alternatively to prove that the loss on some targets can become significantly smaller than the guarantee. This analysis should take into account that by lemma 2 choosing a different value for η' might result in a different guaranteed value of p, which might complicate taking the limit of η' to zero in corollary 4.

3.3.2. Finding the Theorem's Model

Figure 3.1 illustrates the relevance of the theory in the MSA setting, again with the example domains and the simple weighted combiner $h_W(z)$. Theorem 3 guarantees that for some z, $h_W(z)$ achieves a loss that upper bounds the guarantee. In the figure this is seen to be $z = 0.30$, the z that gives the optimal guarantee. The theorem guarantees that at least one such value exists. Thus, to find the optimal guarantee means to find the z for which the theorem's bound holds.

To begin with, this can be done by minimising the difference between the left- and right-hand sides of the bound. An algorithm implementing this optimisation is for specific cases given by [9]. (Further discussion on the algorithm and optimisation problem can be found in appendix B.)

Next, note from figure 3.1 that the optimal guarantee can *not* be found by maximising the loss on the matching mixture, as that maximum has a different z. Nor can it be found by minimising the upper bound, as one might be tempted to do because it upper bounds a loss that is to be minimised. This does not work because by the theorem the bound only holds for some z, as was also seen in the figure.

3.3.3. Value of the Bound and Negative Transfer

The value of the upper bound of theorem 3 is (safe for a small smoothing constant) $L_{\mathcal{D}_p}(h(p))$. One interpretation of this value is the loss on target D_λ where as parameter p the true mixture weights λ are used. Or equivalently: the loss of the model with parameter p on the target mixture with as mixture weights the parameter $\lambda = p$. This mixture is denoted as the "matching mixture" D_λ of the model $h(p)$. Thus the near-optimal guarantee is given by the loss on a model's matching mixture.

Furthermore, recall firstly that the value of the upper bound is unknown and might be large,[1] resulting in a loose, void bound; and secondly recall that as goal of the MSA setting a performance guarantee was chosen to counteract negative transfer. If the bound is void, negative transfer can still occur on some targets $\mathcal{D}_q$, and additionally a void bound is not useful in practice. Therefore, to ensure that the bound is not void and thus useful, it is needed to quantify the value of $L_{\mathcal{D}_p}(h(p))$, i.e. the loss on a matching mixture. To do so it is sufficient to prove that the bound is satisfactory for *any* matching mixture $\mathcal{D}_p$, so that for whichever p is guaranteed by the theorem, the bound will be satisfactory.

This implies the following perspective on the theorem: if a model is known to perform well for all matching mixtures $\mathcal{D}_p$, then by application of the theorem (a parameter value exists for which) the model also performs well on any mixture target $\mathcal{D}_q$.

This is precisely how to quantify the upper bound for the case where distributions are used as additional knowledge, as will follow in chapter 4.

3.4. Conclusion

This chapter has presented a theory from literature that guarantees a robust model exists in the MSA setting. The robustness property provided by the theorem is the goal of the setting and is almost the best (tightest) possible under the current assumptions. The model it guarantees was empirically shown to be the optimal model for one example scenario. Additionally, an algorithm exists that finds this guaranteed model.

However, it is not yet known *how* robust the model is: the bound might be vapid, which would render it useless in practice to counteract negative transfer. The next chapter will quantify the upper bound and show how a model can be proven to be satisfactorily robust. This will be done by using distributions as additional knowledge.

4

Additional Knowledge in the MSA Setting

4.1. Introduction

Previously, chapter 3 has shown the existence of a model that is robust to *any* target mixture and that was empirically shown to be the optimal model in the MSA setting. This was shown to be a fundamental property of the MSA setting by assuming only continuity of the model in its parameter. (And so no model (family) like a specific combiner was assumed.) However, it is not yet known *how* robust the model is: the bound might be void.

This chapter will first show that the upper bound can be quantified by assuming a combiner that uses distributions as additional knowledge of the sources (sec. 4.2). It will be shown that this yields a not-void bound and thus a satisfactory robustness property.

Next, it is hypothesised that other additional knowledge than the distributions can also be used to quantify the guarantee i.e. provide a robust model (sec. 4.3). This will not be proven, but it will be analysed why the distributions work as additional knowledge and so which other knowledge has the potential to improve robustness. Thought experiments will confirm this hypothesis for the case of using either training sample size, model complexity or knowledge of non-i.i.d. sampling as additional knowledge.

This chapter will then have answered the question:

> "What could be used as additional knowledge to ensure satisfactory robustness of a model in the MSA setting?" (RQ3)

4.1.1. Motivating example

Distributions of the source data are informative to the selection of a source model as the target model, and by extension to how much weight to assign them in a combiner.

As a motivating example, consider the domains in figure 4.1. The domains are subsets of the same feature space in which they are far apart. Now, given a new sample at $x_1 = -100$, it is clear that the preferred source classifier to select should be h_1, as it was trained on more representative data. I.e. its training data (around $x = -100$) more resembles the test data (at $x = -100$) than does h_2's training data (around $x = 100$).

By extension, if not a model is to be selected but rather they are to be combined, letting the combining weights depend on the source distributions might improve performance by 'selecting' the model whose training data most resembles the test data point. That is, it addresses the problem of different training and test distributions.

To construct such a combiner, in addition to the source classifiers also the source distributions need to be available as *additional knowledge* of the source domains. The next section will confirm that this *Distribution-Weighted combiner* has favourable properties.

(Similar to knowing the marginal distributions, having class information in the form of e.g. the joint distributions could also be informative and improve performance.)

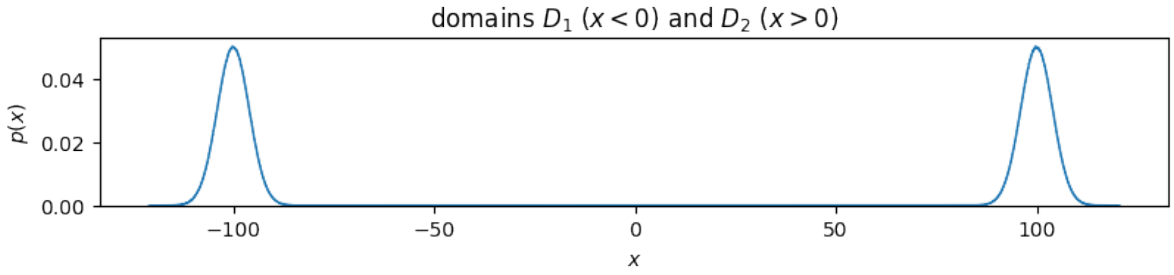


Figure 4.1: Motivating example domains for using distributions as additional knowledge. The domains are subsets of the same feature space in which they are far apart.

4.2. Distribution-Weighted Combiner

4.2.1. Theory

Introduction to Theoretical Analysis

The robustness property will now be quantified and thereby shown to be satisfactory for the distribution-weighted (DW) combiner.

Recall the perspective (sec. 3.3.3) on MSA theory (th. 3): if a model is known to perform well for all matching mixtures $\mathcal{D}_p$, then by application of the theorem for some parameter value the model also performs well on any mixture target $\mathcal{D}_q$. Therefore it is to be shown that the DW-combiner's performance on any matching mixture is satisfactory, i.e. to bound $L_{\mathcal{D}_p}(h_{DW}(p))$ for all p. Lemma 5 does just that, after which corollary 5.1 uses this lemma to quantify the upper bound in theorem 3.

To bound the loss of all matching mixtures (lem. 5) assumptions are required on the form of the target model (a combiner), as well as knowledge of the losses of the (source) models it combines, on the source domains. Specifically, the loss on target $\mathcal{D}_p$ is expressed in terms of $L_{\mathcal{D}_k}(h_k)$ which is known to be upper bounded by ϵ, the worst source loss.

Assumptions and Notation

The combiner is assumed linear, i.e.

$$h_T(x, y; \theta) = \sum_k w_k(x, y; \theta) h_k(x, y) \tag{4.1}$$

with weights w_k that are continuous in parameter $\theta \in \Theta$. (Consequently $h_T(x, y; \theta)$ is continuous in θ.) The weights $w_k \geq 0$ are assumed normalised $\sum_k w_k = 1$ which implies $w \in \Delta$, i.e. they are valid mixture weights. Specifically,

$$w_k^\eta(x, y; z) = \text{normalise}\left(z_k \mathcal{D}_k(x, y) + \eta \frac{\mathcal{U}(x, y)}{K}\right) \tag{4.2}$$

with hyperparameter $\eta > 0$ for smoothing and $\mathcal{U}$ the uniform distribution. Weights $z_k \geq 0$ are normalised as $\sum_k z_k = 1$ which implies $z_k \in \Delta = [0, 1]$. The normalisation is defined for some K-length vector x with $x_k \geq 0\ \forall k$ as $\text{normalise}\,(x_k) = \frac{x_k}{\sum_k x_k}$ where $\sum_k x_k \neq 0$. The normalisation is then always defined because η guarantees the denominator is strictly positive.

With these weights this model is the distribution-weighted (DW) combiner $h_{DW}^\eta(z)$ and it is equivalent to [9]'s combiner. Note firstly that in contrast with the weighted combiner $h_W(z)$ (chapter 2) the combining weights depend on x, y. Secondly, equation 4.2 uses as additional knowledge the distributions $\mathcal{D}_k(x, y)$.

The loss of source models is bounded:

$$\ell\left(h_k(x, y)\right) \leq M \quad \forall k, (x, y)$$

and they are relatively accurate on their own domain; their expected loss is bounded:

$$L_{\mathcal{D}_k}(h_k) \leq \epsilon \quad \forall k$$

Lemma 5. [1] *Consider the distribution-weighted combiner $h^{\eta}_{DW}(z)$ in equation 4.2. For any $\eta > 0$ and $z \in \Delta$ it holds that:*

$$L_{\mathcal{D}_z}(h^{\eta}_{DW}(z)) \leq \epsilon + \eta M$$

Proof. See page 35.

Corollary 5.1. [2] *Consider the distribution-weighted combiner $h^{\eta}_{DW}(z)$ in equation 4.2. There exist a $z \in \Delta, z_k \neq 0$ and $\eta > 0$ such that its loss on any mixture $\mathcal{D}_q$ is bounded by*

$$L_{\mathcal{D}_q}(h^{\eta}_{DW}(z)) \leq \epsilon + \delta \tag{4.3}$$

for any $\delta > 0$.

Proof. Consider the loss of $h^{\eta}_{DW}(z)$ on any mixture $\mathcal{D}_q$. By theorem 3 this loss is bounded by

$$L_{\mathcal{D}_q}(h^{\eta}_{DW}(z)) \leq L_{\mathcal{D}_z}(h^{\eta}_{DW}(z)) + \eta'$$

for any $\eta' > 0$ and some parameter values $z \in \Delta, z_k \neq 0$ and $\eta > 0$ that are guaranteed to exist. Applying lemma 5, which holds for any z so also for the z that is guaranteed to exist here, results in:

$$L_{\mathcal{D}_q}(h^{\eta}_{DW}(z)) \leq \epsilon + \eta M + \eta'$$

Choosing η and η' such that $\eta M + \eta' = \delta$ concludes the proof. For example: $\eta = \frac{\delta}{2M}$ and $\eta' = \frac{\delta}{2}$. □

Existence and Robustness

Similar to the discussion of MSA theory in section 3.2, note the following. The bound only holds for (at least) one model $h(p)$, *not* for the entire family of models $h(p)$ $\forall p$. The existence of that model is guaranteed by theorem 5.1 and the theorem's bound (eqn. 4.3) is a robustness property of the model. To find a model with such a robustness property is the goal of the setting. Contrary to section 3.2, the bound's value is now known: it equals the worst source loss ϵ. (Note also that section 3.3's discussion on tightness and optimisation of the bound still applies here.)

Significance

This is a significant guarantee: a model $h^{\eta}_{DW}(z)$ exists whose loss on *any* mixture $\mathcal{D}_q$ is at most the worst source loss ϵ, i.e. there exists some DW-combiner that guarantees the same performance on *any* target mixture as the worst source model *on its own domain.*

Normalised Target Model

Recall that models h were assumed to have as output for each $x \in \mathcal{X}$ the vector of posterior class probabilities $p(y|x)$, which is required for the cross-entropy loss (sec. 2.2). Although this holds by assumption for the source models, it might not for the target model. That is, it is not guaranteed that for each $x \in \mathcal{X}$: $\sum_{y \in \mathcal{Y}} h_T(x, y; \theta) = 1$. To ensure a valid target model it is normalised as defined for each $x \in \mathcal{X}$ by:

$$\bar{h}_T(x, y; \theta) = \frac{h_T(x, y; \theta)}{\sum_{y \in \mathcal{Y}} h_T(x, y; \theta)} \tag{4.4}$$

Corollary 5.1 still holds for the normalised DW-combiner $\bar{h}_{DW}(x, y; z)$ under mild assumptions. [9, apx.C] This can be proven with the help of the following lemma. It holds for general target models, thereby showing that MSA theory is not significantly affected by normalisation of the target model. It need only be known by how much (ξ) a target model at most violates the normalisation assumption (eqn. 4.5). For the normalised DW-combiner, in corollary 5.1 the chosen η then depends on ξ and the same bound (eqn. 4.3) holds.

[1] This lemma and its proof are mostly equivalent to [9, sec. B.3, cor. 3].
[2] This corollary is mostly equivalent to [9, sec. B.3, cor. 3].

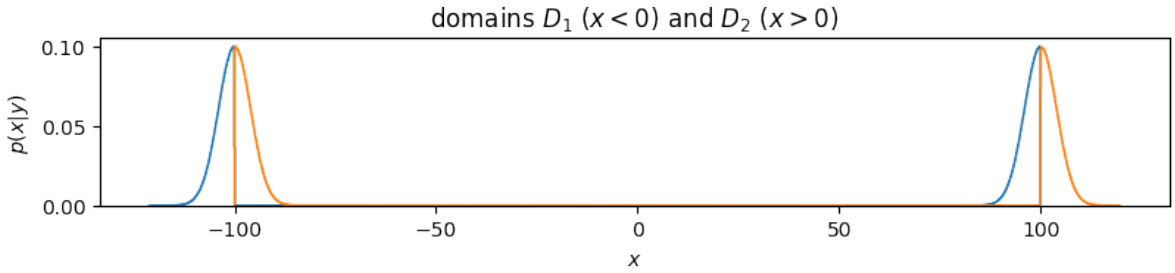


Figure 4.2: Class-conditional distributions of motivating example domains (fig. 4.1) for using distributions as additional knowledge.

Lemma 6. [3] *Consider the normalised combiner* $\bar{h}_T(x, y; \theta)$ *in equation 4.4. If for any* $x \in \mathcal{X}$

$$\sum_{y \in \mathcal{Y}} h_T(x, y; \theta) \leq 1 + \xi \tag{4.5}$$

for some ξ *then the loss on a mixture* $\mathcal{D}_q$ *is bounded by*

$$L_{\mathcal{D}_q}(\bar{h}_T(\theta)) \leq L_{\mathcal{D}_q}(h_T(\theta)) + \xi \tag{4.6}$$

Proof. See page 36.

4.2.2. Experiments

Experimental Setup and Baselines

The experimental setup is the same as in section 2.5.1. As baselines, in addition to the source models h_k the DW-combiner is to be compared with the simple weighted combiner because the latter does not make use of the distributions. So this evaluates the effect of using the distributions.

Results

First recall the motivating example for using distributions as additional information (fig. 4.1). Figure 4.2 shows class-conditionals for the same domains. As is also clear from the class-conditionals, h_1 always misclassifies half of the samples in D_2, whereas h_2 attains the Bayes error rate on its own domain (and vice versa). This is evident in the robustness of models when applied on these domains: the best source model guarantees an expected loss of at most 17.28, whereas the DW-combiner guarantees $L = 0.14$, which agrees with MSA theory's guarantee of $\epsilon = 0.14$. Thus this shows MSA theory for the DW-combiner in practice, and shows that it possibly yields large improvements in robustness, the goal of the setting.

The DW-combiner is also analysed for the example domains of figure 2.1, where the domains differ by a distributional shift. Experiments similar to those in section 2.5 give rise to the following three analyses.

Firstly, the robustness guaranteed by MSA theory is compared with the robustness of the source models as baselines (fig. 4.3). The theorem guarantees a model that is as robust as the worst source model performs *on its own domain*. Indicated as ϵ in the figure, the theory guarantees improved robustness compared with the source models. Indeed note well and recall from section 2.5 that the source models themselves are not robust at all, a behaviour that was denoted as negative transfer. Figure 4.3 illustrates this: the most robust source model guarantees a loss of at most 1.05 on any mixture target D_λ, whereas MSA theory ensures a DW-combiner exists that guarantees a loss of at most $\epsilon = 0.66$. Thus MSA theory guarantees a significant robustness improvement for some DW-combiner.

[3] This lemma's proof reformulates a part of [9, sec. C, th. 5].

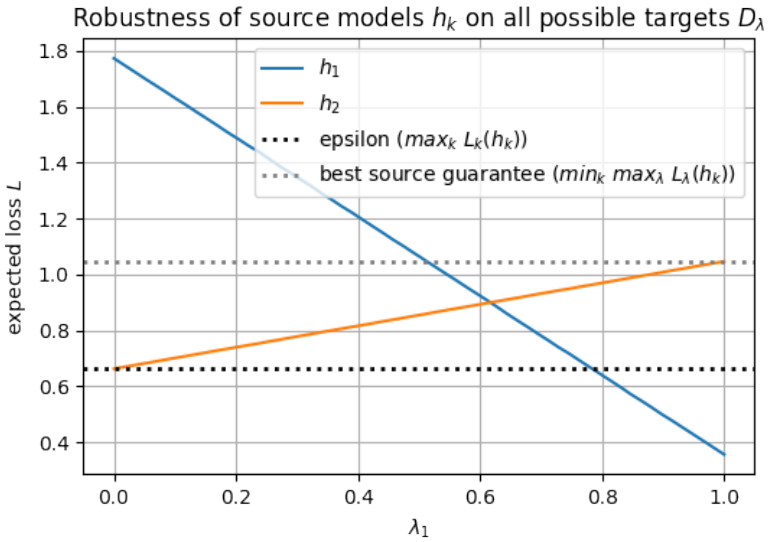


Figure 4.3: Comparison of the robustness guaranteed by MSA theory and the robustness of the source classifiers. The goal of the setting is to find the most robust model, on which the theory's model outdoes the best source model h_2. (This figure continuous figure 2.2 so also uses linear source classifiers h_k, the domains in figure 2.1, a sample size of domains of $N = 2.5 \cdot 10^6$ and a gridsize for λ of 60.)

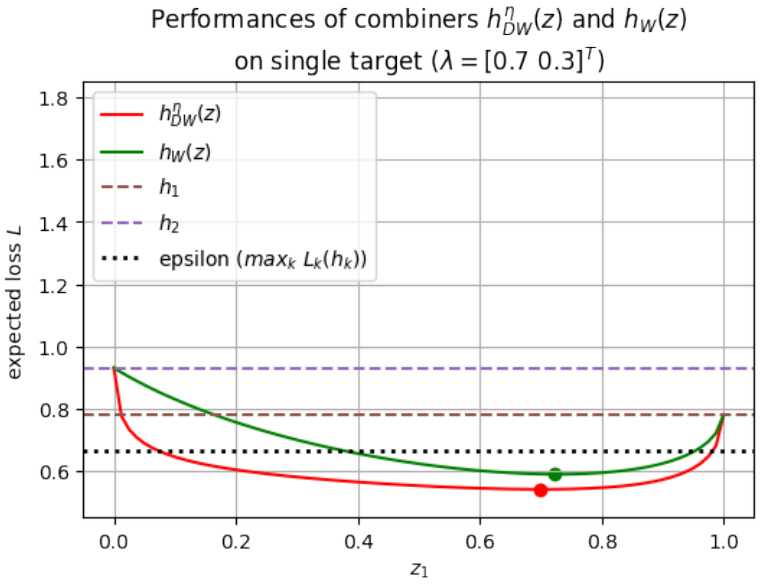
Secondly, the assumed form of the target model (the DW-combiner) is evaluated on a single target (fig 4.4a). It is seen that on this target mixture for all z the DW-combiner h^{η}_{DW} has a lower loss than the simple weighted combiner h_W. Furthermore, the optimal DW-combiner has a lower loss than h_W as well as both source models h_k. Thus using the source distributions as in the DW-combiner can improve performance.

Lastly, the robustness of the DW-combiner is evaluated and compared with the robustness guaranteed by MSA theory (fig. 4.4). This empirically shows that the optimal DW-combiner achieves a robustness of ϵ, exactly as guaranteed by MSA theory.

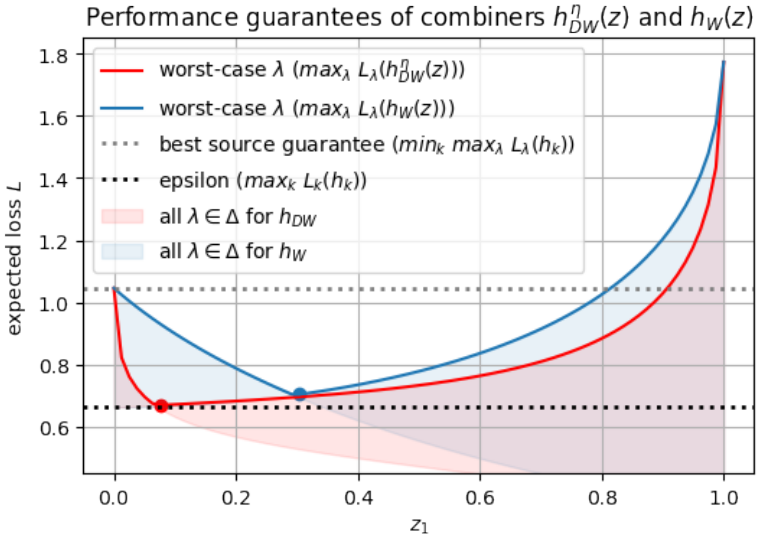
This robustness is better than that of the combiner that does not use the distributions, confirming that in some scenarios relying on the distributions to combine can improve robustness. That is, distributions as additional knowledge can improve performance, and therefore this shows that additional knowledge can improve robustness.

4.2.3. Conclusion

MSA theory from chapter 3 was applied to the distribution-weighted combiner, yielding a guaranteed robust model. This robustness of ϵ was shown to be a significant result and improve on both the source models and the weighted combiner that does not use distributions. Since using distributions as additional knowledge can improve robustness, additional knowledge can improve robustness. The next section will investigate which other knowledge might be used in that manner.



(a) Single target with $\lambda = [0.7\ 0.3]^T$. This figure evaluates the choice of the combiner.



(b) Worst-case target $\lambda = \arg\max_\lambda\ L_\lambda(h(z))$ i.e. robustness. This figure evaluates the DW-combiner's robustness as compared with MSA theory's guarantee.

Figure 4.4: Comparison of all (distribution-)weighted combiners h^{η}_{DW} and $h_W(z)$ (equations 4.2 and 2.2 for all z) with linear source classifiers h_k when used on either a single target (fig. 2.3a) or the worst-case target (fig. 2.3b). The performance on the worst-case target is a performance guarantee and minimising it is the goal of the setting. (This figure continuous figure 2.3 so also uses linear source classifiers h_k, the domains in figure 2.1, a sample size of domains of $N = 2 \cdot 10^6$ and a gridsize for λ of 80.) For the DW-combiner the hyperparameter used for smoothing was $\eta = 0.001$.

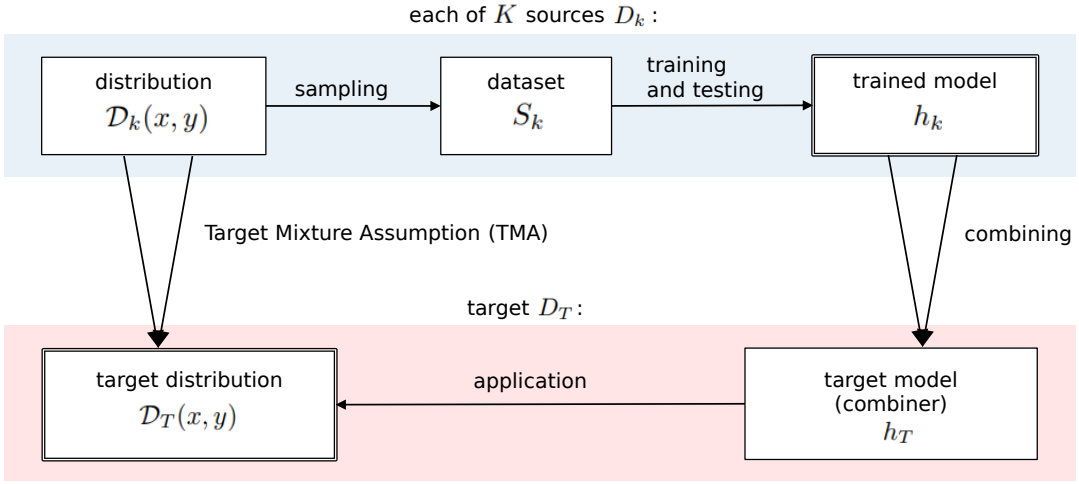


Figure 4.5: Visualisation of the process of constructing a target model in the MSA setting. Source models (top) are trained with the goal that h_k performs well on (true) distribution $\mathcal{D}_k$. The MSA setting concerns the target (bottom), where the goal is performance of a target model as expected on the (worst) target distribution. (Recall from chapter 1 that there is no 'target test set' (no target data is available) but only a target distribution.) The aim is to design the combiner such that its application on the target distribution performs best. That is, application of source models on other domains than they have been trained on.

4.3. Other Additional Knowledge

4.3.1. Motivation

Upto now it was shown that additional knowledge of the source domains can improve performance in the MSA setting. Specifically, it was shown how distributions can be used in a weighted combiner to ensure a sufficiently robust combining model. This motivates the question what other knowledge might be used in that manner.

To that end, it is now analysed why distributions worked as additional knowledge.

As a guide, figure 4.5 details the construction of a target model in the MSA setting. Suppose first that h_k is perfectly trained for performance on $\mathcal{D}_k$, in the sense that it finds the Bayes classifier. If h_k is now applied (through the combiner) on a different distribution $\mathcal{D}_T$ this violates the implicit machine learning assumption that training and test data are similarly distributed. To account for this, the combiner might utilise knowledge of the source distributions.[1][9]

So, additional knowledge *of the trained source model* is used to correct any mismatch between that trained model (what for and how it was trained) and on what it is applied. As seen from the figure, distributions are not the only possible mismatch: everything related to the training procedure (logically) influences its performance. Any knowledge of the source models' training procedures therefore might help decide how much weight to give the model in the combiner. For example, current methods do not account for the training sample size, but if model A is trained with 1000 times more samples it might be preferred to model B, even though model B has a higher *fraction* of their samples at the new samples feature.

Any additional knowledge is available in addition to the trained source models, which are always available in the MSA setting.

To be precise, the goal is to optimise for robustness (sec. 2.4), which is a model's loss on the data's underlying ('true') distribution. Anything of the training procedure that influences this loss (e.g. ϵ) therefore influences the robustness.

Based on this observation, possible candidates for additional knowledge in this MSA setting will now be hypothesised. Next, some of those candidates for additional knowledge are discussed by example. Finally, a discussion of these examples will outline some requirements for combiners that use additional knowledge.

4.3.2. Candidates for Additional Knowledge

Recall and note well that in the setting only additional knowledge of the source domains might be available, never of the target domain. In fact, the target domain is unknown.

Quantifiable

A candidate for additional knowledge can be anything that influences the performance of the final trained source model h_k on a fixed (target) distribution. Importantly, it must be quantifiable so that it can be used in a target model; e.g. some human decision-making processes in model selection or data acquisition (e.g. manual discarding of bad samples) can not be candidates.

Need Not Be Changeable

It is not needed to be able to change the candidate's value, only to be able to measure it. For example, any sampling biases need not be resolved, only their values need to be known to be able to account for them.

Example Candidates

Recall that anything of the training procedure could influence robustness so is a candidate for additional knowledge. To clarify this, some examples are now listed, guided by a visual depiction of a target model's construction in the MSA setting (fig. 4.5).

A dataset is (modelled as being) sampled from a (true) distribution $\mathcal{D}(x, y)$, of which the following candidates can be distinguished:

- Marginal distribution $\mathcal{D}(x)$
- Class-conditional distribution $\mathcal{D}(x|y)$
- Class prior distribution $\mathcal{D}(y)$
- Joint distribution $D(x, y)$

The sampled dataset is processed before it is used for training and testing. There the final trained model might be influenced by for example:

- Data acquisition: measuring equipment biases, non-i.i.d. sampling
- Data preparation: data conversion (e.g. rounding of numbers, non-reversible feature transformations

On this dataset training is performed within some model evaluation procedure.

- Loss on test set (estimated true loss) ϵ: mean value, variance (as influenced by e.g. N_{test})
- Loss on training set (apparent loss): ditto
- Overfitting: difference between losses on test and training sets
- Number of samples N_{train}, N_{test}

Those models are trained with some algorithm, yielding candidates such as:

- Hypothesis space $\mathcal{H}$ (before training): VC-dimension or other measures of model complexity
- Effective hypothesis space (when trained, for example due to regularisation, early stopping, ...): VC-dimension or other measures of (the trained) model complexity. (An example of model misspecification follows in the next section.)
- For SVMs: margin

Many others are possible depending on algorithms and training procedures used.

4.3.3. Thought Experiments

Thought experiments will now illustrate some candidates for additional source knowledge, specifically the training sample size, non-i.i.d. sampling and model misspecification.

Experimental Setup and Baselines

The experimental setup and baselines are the same as in section 4.2.2. Notably, a combiner that uses additional knowledge is to be compared with the same combiner that does not use that knowledge. This allows an evaluation of the effect that using additional knowledge has. For example, section 4.2.2 compared the DW-combiner with the simple weighted combiner, because the former reduces to the latter if all distributions $\mathcal{D}_k$ used in the combining weights are the same.

Model Complexity

Consider the following example of model misspecification. Domains D_1 and D_2 have the same joint distribution ($\mathcal{D}_1(x, y) = \mathcal{D}_2(x, y)$) that requires a quadratic decision boundary to achieve the Bayes loss. Let source model h_1 be quadratic discriminant analysis and h_2 be linear discriminant analysis, i.e. h_1 is correctly specified and h_2 is misspecified. Then the losses of the source models on the true distribution $\mathcal{D}_1 = \mathcal{D}_2$ will be related by $\epsilon_2 > \epsilon_1$ and by theorem 5.1 the performance guarantee is $\epsilon = \epsilon_2$. However, in this scenario h_1 would always perform better than h_2, achieving an improved robustness of ϵ_1. Furthermore, the difference between h_1 and h_2 is quantifiable, for example by using the order of the trained model's polynomial decision boundary as a measure of model complexity. So in this example, if the combiner additionally had the policy "if distributions are equal, then prefer the most complex model", the combiner's robustness would improve to ϵ_1 as a result of using additional knowledge.

Note also that in this scenario the best target model for a source domain is not its own source model.

Example of Non-i.i.d. Sampling

Consider the following example of differences in sampling between domains, for example a local difference in training sample size. Domains D_1 and D_2 are subsets of a single feature and label space and in a region $R \subset \mathcal{X} \times \mathcal{Y}$ have the same joint distribution. Training sample sizes are large for both domains, but $\mathcal{D}_2$ is not sampled i.i.d., instead yielding the estimate $\hat{\mathcal{D}}_2$. This leads to a different prior $p(y)$ to be estimated from the data, influencing the posterior $p(y|x)$ and as a result h_2 might not learn to classify points in region R correctly and $\epsilon_2 > \epsilon_1$. That is, source model h_2 is optimised for $\hat{\mathcal{D}}_2$ instead of $\mathcal{D}_2$ and so ϵ_2, the loss on true distribution $\mathcal{D}_2$, increases.

However, in this scenario h_1 will perform better than h_2 on any target mixture, achieving an improved robustness of ϵ_1. This worse robustness is the result of the non-i.i.d. sample, which is quantifiable, for example if a distribution is known that specifies the probability of a sample being drawn at (x, y). If in this example a DW-combiner is used that additionally corrects for the non-i.i.d. sampling, the combiner's robustness would improve to ϵ_1 as a result of using additional knowledge.

Example of Training Sample Size

If the only difference between domains is their distributions and sample size, the model trained on the most samples is preferred for minimal error on their true distributions. This is because fewer samples means a larger variation in possible distributions of the training set $\hat{\mathcal{D}}$, and the more $\hat{\mathcal{D}}$ differs from the true distribution $\mathcal{D}$, the larger the true error ϵ. So, though a smaller training set can also be a lucky sample, in expectation over all possible sampled training sets of that size it is best to select the model trained on the most samples.

Consider as an extreme example two domains with the same basic 1D distribution $\mathcal{D}_1 = \mathcal{D}_2$ that is $p(x|y = -1) \sim \mathcal{N}(-1, 1)$ and $p(x|y = 1) \sim \mathcal{N}(1, 1)$ so that the Bayes classifier has its decision boundary at $x = 0$. Training sample size is infinite for D_1 and finite and small for D_2. Then a linear model h_1 is the Bayes optimal classifier and h_2 has its decision boundary close to but not exactly at $x = 0$. This implies $\epsilon_2 > \epsilon_1$, so that h_1 is always more robust than h_2.

4.3.4. Discussion

Improvement Upon Current Method

Note that in none of the examples the DW-combiner is optimal and in all examples it is described how it would improve if additional knowledge is taken into account.

Requirements For Combiners That Use Additional Knowledge

Recall the assumptions in section 2.2. A target model that uses additional knowledge should satisfy the assumptions so that MSA theory can be used to prove robustness against any target mixture, as in section 4.2.1.

One requirement that might sound intuitive is best *not* made: the combiner should *not* be required to reduce to the source model on that source's distribution, i.e. for $z = \lambda_k$. This is because the best target model for a source domain is not always its own source model, as was observed in the examples.

Discussion on Bound in Practice

Recall that the goal is to optimise for robustness (sec. 2.4), which is a model's loss on the data's underlying ('true') distribution. Anything of the training procedure that influences this loss (e.g. ϵ) therefore influences the robustness.

Note that the DW-combiner uses the true distributions, which are not changed by the training process. For example, changing the training sample size does not change the combining weights, but it does change ϵ and thus the DW-combiner's bound. This is as hypothesised and was observed in each example.

In practice, however, care should be taken that such a worse performance does not go unnoticed. Consider for example the non-i.i.d. sampling example. If it is not realised that the sample of $\mathcal{D}_2$ is not i.i.d., then $\hat{\mathcal{D}}_2$ will erroneously be interpreted as the true distribution, and as a consequence the bound from theorem 5.1 is ϵ as estimated on $\hat{\mathcal{D}}_2$. That bound holds for targets that are a mixture of $\mathcal{D}_1$ and $\hat{\mathcal{D}}_2$. However, in truth targets are mixtures of $\mathcal{D}_1$ and $\mathcal{D}_2$, leading to a violation of the erroneous bound. [8, Th.14] has shown that the actual bound in this situation includes a divergence factor $d(\mathcal{D}_2||\hat{\mathcal{D}}_2)$, making it looser.

Furthermore, note that for the goal of best guarantee, overfitting is not an issue because ϵ is the loss on the true distribution, for which a lower value is always better. The effect of overfitting is an increased ϵ, leading to a (known) looser bound.

4.3.5. Conclusion

Additional source knowledge other than the distributions was motivated to be able to improve robustness in the MSA setting. Specifically, since the robustness is an expected loss on an underlying ('true') distribution, any influence of the training process of the source models that influences the true loss is a candidate for additional knowledge. In addition the candidate should be quantifiable, but it need not be changeable. A non-exhaustive list of candidates was presented, of which three were investigated by thought experiments: model complexity, non-i.i.d. sampling and training sample size. For each example it was reasoned why the DW-combiner is insufficient and that using that type of additional knowledge should improve performance in the example scenario.

Lastly, a brief discussion on using combiners with additional knowledge in practice warned that MSA theory's bound might be unwittingly loose in practice if models are not trained for their true distributions, for example due to being unaware of any non-i.i.d. sampling.

5

Discussion and Conclusion

5.1. Conclusion

This work has formulated the main question

> "How can pre-trained models be combined for performance on all their source domains, without access to data?" (RQ)

as a problem in the Multi-Source Domain Adaptation (MSA) setting.

The contributions are threefold. To begin with, the MSA setting and theory were explained and illustrated. A disadvantage of the setting was seen to be non-robustness (negative transfer) of the target model, though MSA theory guaranteed that a robust model does exist. As a second contribution, existing MSA theory's assumptions were clarified and the theory was split in two. One half was inherent to the MSA setting and yielded the loss on a matching mixture as the robustness property for a guaranteed model. The other half depended on the additional source knowledge used and was presented as a perspective on existing MSA theory. Thirdly, it was investigated what makes useful source knowledge in the MSA setting and how it could be used to derive alternative MSA theory. Source knowledge was reasoned to be useful if it is quantifiable and affects the true source loss, and is worthwhile to investigate because using it can improve upon the DW-combiner. Additional knowledge can be used to derive alternative MSA theory by bounding the loss on the matching mixture for a family of combining models that use that additional knowledge.

In conclusion the main research question is answered as follows. By modelling the problem in the MSA setting, MSA theory can be applied to yield a guaranteed to exist model with a robustness property. Finding the most robust model is the goal of the setting. By using additional knowledge of the source domains the robustness is proven to be satisfactory.

5.2. Related Work

The MSA setting and theory in this work are most similar to those for the "probability model" in [9]. Some of their assumptions are more general; they for example do not assume that models output the posterior distribution, simply a distribution. However, this work expands on [9] by considering other additional knowledge than the distributions they use in their distribution-weighted combiner.

Another difference with [9] is that this work has motivated and defined the goal of the MSA setting (sec. 1.1.2, 2.4). By contrast, [9] states MSA theory and optimises its bound directly. The goal of minimising a performance guarantee is also different than what [9] optimises for (see section 3.3.2), although it was empirically shown (ch. 3) that [9] finds the same optimum. Furthermore, some theory was stated differently. A clear example is the distinction this work makes between setting-specific and additional knowledge-specific theory. Another example is a focus on mixtures over sources (see section 3.2) which arguably results in cleaner, more comprehensible theory. As an example, compare lemma 2 and theorem 3; here the focus on mixtures allowed the interpretation of the matching mixture.

5.2.1. Related Settings

Some different but related DA settings are now characterised to place MSA in their context. For literature on related but different MSA settings the interested reader is referred to [9, apx.G].

Multiple-Source Domain Adaptation (MSA) (as discussed in this work): has no access to any source or target data, but available are pre-trained source models and optionally additional source knowledge such as their distributions. The target is unknown but constrained to be a mixture of the sources. [1][8][9]

Multi-Domain Learning: each sample of a single dataset also has a domain indicator and the goal is to generalise to all data. [5]. This is equivalent to splitting the data into source domains and knowing the target is a uniform mixture of sources. Thus labelled source (and target) data is available for training.

Domain Adaptation: typically has access to unlabelled target data and labelled source data during training. [10] The target is known, allowing adaptation for the specific target.

Domain Generalisation: has access to labelled source data during training but, like MSA and unlike DA, has no target data or other knowledge. [14] The goal is to generalise to unknown target domains.

Transfer Learning: a broad category that includes all DA settings as well as allows transfer between different learning tasks. The common denominator is that training in the target domain uses source knowledge.

5.3. High-Level Overview of MSA Theory

For different types of additional knowledge MSA theory could yield a useful result such as corollary 5.1. To aid the derivation of similar theory using other additional knowledge, a high-level overview of MSA theory's assumptions is now presented.

The learning process in the MSA setting (as depicted in figure 4.5) starts with source models h_k trained on source domains D_k. It is desired to construct a target model h_T that is optimised for a target domain D_T. To do so, two assumptions are made. Firstly, by defining the relationship between source and target *domains* (i.e. the TMA), the source and target *loss* are related for any model (cor. 1). In the MSA setting, it turns out that this implies (for a guaranteed model) a relationship between the loss on any mixture and on that model's matching mixture (th. 3). Secondly, by defining the relationship between source and target *models* (i.e. a combiner), the loss on a matching *mixture* is related to the loss on the *sources* (lem. 5). All together this yields (for a guaranteed target model) a relationship between the loss on any target mixture and the source loss of source models. Thus the target model's target loss is expressed in terms of source variables only, which are known in the setting. The robustness of the target model is then expressed in known source variables only.

An implication is that for satisfactory robustness additional knowledge should focus on reducing the loss on any matching mixture. That is, then knowledge has a good chance to yield a satisfactorily robust target model.

Lastly, keep in mind that using any additional knowledge only helps in scenarios where that knowledge is different between the domains. If for example a DW-combiner is used in a scenario where the source distributions are the same, there are no benefits to using it over a simple weighted combiner that does not use the distributions.

5.4. Lessons From This Work

Some take-aways from this work now follow.

The goal of performance on all source domains can be modelled as performance on an unknown mixture target. This is a performance guarantee and a robustness property.

MSA theory addresses the main issue of the MSA setting, i.e. non-robustness. The loss on a matching mixture plays a central role.

Robustness in the MSA setting can be improved by basing the combiner on additional knowledge of the source domains. That combiner's robustness can be proven to be satisfactory by evaluating its loss on any matching mixture and applying MSA theory.

5.5. Future Work

The focus of this work has been the perspective of additional knowledge in MSA theory. Recall that it has been discussed which additional knowledge is useful, what its properties should be and how MSA theory can be used to evaluate its robustness. A logical next step is to formulate combiners that use this knowledge and to derive MSA theory for them, i.e. to bound their loss on their matching mixtures, specifically for the discussed knowledge: model complexity, non-i.i.d. sampling and training sample size.

The current approach to finding the target model for a scenario is to find its optimal parameters by optimisation of the bound in theorem 3. Prior to this, a specific combiner such as the DW-combiner is selected using what might be called a model selection procedure. It might be beneficial to incorporate knowledge of the bound in the model selection procedure. For example, the model could be selected from various combiners based on their expected loss on the matching mixture.

MSA theory's statement is the result of a fixed-point mapping (FPM) assumed in its proof. Using a different FPM might therefore yield new MSA theory either for the current MSA setting, or for alternative settings that use a different TMA. Appendix C discusses this further.

An alternative MSA setting might be formulated where sources are themselves allowed to be mixtures. Let's elaborate on this. The setting might be viewed from the following perspective: the loss on any mixture is of interest (i.e. the goal) and the loss on some mixtures is given. Currently those given mixtures happen to be extreme mixtures, i.e. they include only one domain, and it is called a source domain. We hypothesise that this is not necessary and it is sufficient to know the mixture parameter of a source. (Presumably, to prove an updated theorem 3, some criterion is needed to ensure those source mixtures 'span' all domains, similar to how in linear algebra a set of vectors must span a vector space to form a basis for it.)

This alternative MSA theory would cover a larger space of target domains than current MSA theory, as is now illustrated with an example. Consider the artificial problem by [8, sec.6], where three domains are constructed as uniform mixtures of four gaussians $\{g_1, g_2, g_3, g_4\}$: source distributions $\mathcal{D}_1$ of $\{g_1, g_2, g_3\}$ and $\mathcal{D}_2$ of $\{g_1, g_3, g_4\}$. As mentioned by the authors, a uniform mixture $\mathcal{D}_P$ of all four gaussians is not a mixture of the sources $\mathcal{D}_1$ and $\mathcal{D}_2$, i.e. no λ exists for which $\lambda\mathcal{D}_1 + (1-\lambda)\mathcal{D}_2 = \frac{1}{3}(g_1 + \lambda g_2 + g_3 + (1-\lambda)g_4) = \mathcal{D}_P$. If however instead of $\mathcal{D}_1$ and $\mathcal{D}_2$ the gaussians $g_1, ..., g_4$ are considered as sources, then $\mathcal{D}_P$ would be a mixture of sources and updated MSA theory would guarantee a tighter bound than current MSA theory (by [8, Th.2]).

Note that in this alternative setting of each source it is needed to know which mixture of the domains it is, e.g. in this example $\lambda_1 = [\frac{1}{3}\ \frac{1}{3}\ \frac{1}{3}\ 0]^T$ and $\lambda_1 = [\frac{1}{3}\ 0\ \frac{1}{3}\ \frac{1}{3}]^T$. By contrast, in the current MSA setting it is always true that e.g. $\lambda_1 = [1\ 0]^T$ and $\lambda_2 = [0\ 1]^T$.

A

Derivations and Proofs

Theorem 7 (Brouwer's Fixed-Point Theorem)**.** *For any compact and convex non-empty set $C \subset \mathbb{R}^K$ and any continuous function $f : C \to C$, there is a point $x \in C$ such that $f(x) = x$. [9]*

Lemma 2. [1] [2] *Consider K sources $\mathcal{D}_k$. There exists a $p \in \Delta, p_k \neq 0$ such that for any source k*

$$L_{\mathcal{D}_k}(h(p)) \leq \sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta' \tag{3.1}$$

for a model $h(p)$ that is continuous in its parameter $p \in \Delta$ and for any $\eta' > 0$.

Proof. Consider the mapping $\Phi(p) : \Delta \to \Delta$ defined as

$$[\Phi(p)]_k = \text{normalise}\left(p_k L_{\mathcal{D}_k}(h(p)) + \frac{\eta'}{K}\right) = \frac{p_k L_{\mathcal{D}_k}(h(p)) + \frac{\eta'}{K}}{\sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta'} \tag{A.1}$$

where the normalisation is defined for some K-length vector x with $x_k \geq 0 \; \forall k$ as $\text{normalise}\,(x_k) = \frac{x_k}{\sum_k x_k}$ where $\sum_k x_k \neq 0$. The normalisation is always defined because $\eta' > 0$ guarantees the denominator is strictly positive. Note that by assumption $L_{\mathcal{D}}(h(p))$ is continuous in p so Φ is as well, and that Δ is a set such as required for theorem 7, making Φ a valid fixed-point mapping according to that theorem.

There then exists a $p \in \Delta$ such that $\Phi(p) = p$ or equivalently for that p and any k

$$p_k = \frac{p_k L_{\mathcal{D}_k}(h(p)) + \frac{\eta'}{K}}{\sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta'}$$

where $p_k \neq 0$. (To see this, observe that since $\eta' > 0$, $p_k = 0$ yields the contradiction $0 = \frac{1}{K} > 0$.) Rewriting yields

$$\begin{aligned} L_{\mathcal{D}_k}(h(p)) &= \sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta' - \frac{\eta'}{K p_k} \\ &\leq \sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta' \end{aligned} \tag{A.2}$$

which is bounded by discarding the last term. This concludes the proof. □

[1] For $h(p)$ the DW-combiner $h_{DW}(z)$ of equation 4.2 this lemma and its proof are mostly equivalent to [9, sec. B.3, lem. 6] which in turn is based on theorem [1, lem. 2]. Specific equivalences or differences are mentioned when they're relevant.

[2] A more general version of this lemma (for other fixed-point mappings) is given in appendix C as lemma 8.

Theorem 3. *Consider two mixtures $\mathcal{D}_p$ and $\mathcal{D}_q$. There exists a $p \in \Delta$, $p_k \neq 0$ such that for any $q \in \Delta$*

$$L_{\mathcal{D}_q}(h(p)) \leq L_{\mathcal{D}_p}(h(p)) + \eta' \tag{3.2}$$

for a model $h(p)$ that is continuous in its parameter $p \in \Delta$ and for any $\eta' > 0$.

Proof. Take lemma 2 where we apply lemma 1 (using the TMA) to its right-hand side. It follows that there exists a $p \in \Delta, p_k \neq 0$ such that for a model $h(x, y; p)$ continuous in $p \in \Delta$ and for any source k

$$\begin{aligned} L_{\mathcal{D}_k}(h(p)) &\leq \sum_{j=1}^{K} p_j L_{\mathcal{D}_j}(h(p)) + \eta' \\ &= L_{\mathcal{D}_p}(h(p)) + \eta' \end{aligned} \tag{A.3}$$

Now consider the loss of the same model on another mixture $\mathcal{D}_q$:

$$L_{\mathcal{D}_q}(h(p)) = \sum_k q_k L_{\mathcal{D}_k}(h(p)))$$

which by lemma 1 (using the TMA) holds for any p, so also for the p that is guaranteed to exist in equation A.3. So that equation can be used to obtain

$$\leq \sum_k q_k \left(L_{\mathcal{D}_p}(h(p)) + \eta' \right)$$

making the term between brackets independent of k. Finally, because $\sum_k q_k = 1$ by definition, this equals

$$= L_{\mathcal{D}_p}(h(p)) + \eta'$$

This concludes the proof. □

Lemma 4. *In theorem 3, the tightness of the bound in equation 3.2 (i.e. the difference between its left- and right-hand side) is bounded by:*

$$\frac{\eta'}{K} < L_{\mathcal{D}_p}(h(p)) + \eta' - L_{\mathcal{D}_q}(h(p)) \leq \frac{\eta'}{K} \sum_{k=1}^{K} \frac{1}{p_k} \tag{3.3}$$

Proof. First, adapt the proof of lemma 2 by not bounding equation A.2, resulting in that equation's equality instead of the inequality of equation 3.1. Next, adapt the proof of theorem 3 so that it uses the adapted version of lemma 2, resulting in the same statement as theorem 3 except for the bound (eqn. 3.2), which is replaced by the following equality:

$$L_{\mathcal{D}_q}(h(p)) = L_{\mathcal{D}_p}(h(p)) + \eta' - \frac{\eta'}{K} \sum_{k=1}^{K} \frac{q_k}{p_k} \tag{A.4}$$

Define the tightness of theorem 3's bound as:

$$\tau = L_{\mathcal{D}_p}(h(p)) + \eta' - L_{\mathcal{D}_q}(h(p))$$

which by equation A.4 becomes:

$$\tau = \frac{\eta'}{K} \sum_{k=1}^{K} \frac{q_k}{p_k}$$

By the theorem's statement this holds for some $p \in \Delta, p_k \neq 0$ and for any $q \in \Delta$. Therefore the tightness is lower bounded by using $p_k < 1$ and upper bounded by using $q_k \leq 1$, yielding:

$$\frac{\eta'}{K} < \tau \leq \frac{\eta'}{K} \sum_{k=1}^{K} \frac{1}{p_k}$$

This concludes the proof. □

Lemma 5. [3] *Consider the distribution-weighted combiner $h^{\eta}_{DW}(z)$ in equation 4.2. For any $\eta > 0$ and $z \in \Delta$ it holds that:*

$$L_{\mathcal{D}_z}(h^{\eta}_{DW}(z)) \leq \epsilon + \eta M$$

Proof. By the definition of expected loss and the general weighted combiner of equation 4.1:

$$L_{\mathcal{D}_\theta}(h_T(\theta)) = \sum_y \int_{-\infty}^{\infty} \mathcal{D}_\theta(x,y) \ell \left(\sum_k w_k(x,y;\theta) h_k(x,y) \right) dx$$

Since by assumption ℓ is convex in h and $w \in \Delta$ Jensen's inequality can be used, resulting in

$$\begin{aligned} L_{\mathcal{D}_\theta}(h_T(\theta)) &\leq \sum_y \int_{-\infty}^{\infty} \mathcal{D}_\theta(x,y) \sum_k w_k(x,y;\theta) \ell\left(h_k(x,y)\right) dx \\ &= \sum_k \sum_y \int_{-\infty}^{\infty} \mathcal{D}_\theta(x,y) w_k(x,y;\theta) \ell\left(h_k(x,y)\right) dx \end{aligned}$$

where for well-behaved h and w the integral and sums could be interchanged. For the combiner $h^{\eta}_{DW}(z)$ (i.e. using the weights in equation 4.2 with parameter $\theta = z$) this becomes, by applying the TMA in the denominator of the weights,

$$L_{\mathcal{D}_z}(h^{\eta}_{DW}(z)) \leq \sum_k \sum_y \int_{-\infty}^{\infty} \mathcal{D}_z(x,y) \frac{z_k \mathcal{D}_k(x,y) + \eta \frac{\mathcal{U}(x,y)}{K}}{\mathcal{D}_z(x,y) + \eta \mathcal{U}(x,y)} \ell\left(h_k(x,y)\right) dx$$

Since $\frac{\mathcal{D}_z(x,y)}{\mathcal{D}_z(x,y)+\eta\mathcal{U}(x,y)} \leq 1$, this can be bounded by

$$\begin{aligned} &\leq \sum_k \sum_y \int_{-\infty}^{\infty} \left(z_k \mathcal{D}_k(x,y) + \eta \frac{\mathcal{U}(x,y)}{K} \right) \ell\left(h_k(x,y)\right) dx \\ &= \sum_k \left(z_k L_{\mathcal{D}_k}(h_k) + \frac{\eta}{K} \sum_y \int_{-\infty}^{\infty} \mathcal{U}(x,y) \ell\left(h_k(x,y)\right) dx \right) \end{aligned}$$

which implies, because $L_{\mathcal{D}_k}(h_k)$ and $\ell(h_k)$ are bounded by assumption and $\sum_k z_k = 1$,

$$\begin{aligned} &\leq \sum_k \left(z_k \epsilon + \frac{\eta}{K} \sum_y \int_{-\infty}^{\infty} \mathcal{U}(x,y) M \, dx \right) \\ &= \epsilon + \eta M \end{aligned}$$

concluding the proof. □

[3] This lemma and its proof are mostly equivalent to [9, sec. B.3, cor. 3].

Lemma 6. [4] *Consider the normalised combiner $\bar{h}_T(x, y; \theta)$ in equation 4.4. If for any $x \in \mathcal{X}$*

$$\sum_{y \in \mathcal{Y}} h_T(x, y; \theta) \leq 1 + \xi \tag{4.5}$$

for some ξ then the loss on a mixture $\mathcal{D}_q$ is bounded by

$$L_{\mathcal{D}_q}(\bar{h}_T(\theta)) \leq L_{\mathcal{D}_q}(h_T(\theta)) + \xi \tag{4.6}$$

Proof. By equation 4.4 and the definition of the cross-entropy loss:

$$\begin{aligned} L_{\mathcal{D}_q}(\bar{h}_T(x, y; \theta)) &= L_{\mathcal{D}_q}\left(\frac{h_T(x, y; \theta)}{\sum_{l \in \mathcal{Y}} h_T(x, l; \theta)}\right) \\ &= L_{\mathcal{D}_q}(x, y; h_T(\theta)) + \mathbb{E}_{x \sim \mathcal{D}_q(x)}\left\{\log \sum_{l \in \mathcal{Y}} h_T(x, l; \theta)\right\} \end{aligned}$$

where the expectation is over only x and not (x, y) because the log-term is independent of y. By assumption (eqn. 4.5) this is bounded by:

$$L_{\mathcal{D}_q}(\bar{h}_T(x, y; \theta)) \leq L_{\mathcal{D}_q}(x, y; h_T(\theta)) + \mathbb{E}_{x \sim \mathcal{D}_q(x)}\left\{\log(1 + \xi)\right\}$$

and because the logarithm is independent of x,

$$\begin{aligned} L_{\mathcal{D}_q}(\bar{h}_T(x, y; \theta)) &= L_{\mathcal{D}_q}(x, y; h_T(\theta)) + \log(1 + \xi) \\ &\leq L_{\mathcal{D}_q}(x, y; h_T(\theta)) + \xi \end{aligned}$$

where the inequality $\log(a) \leq a - 1$ was used. This completes the proof. □

[4] This lemma's proof reformulates a part of [9, sec. C, th. 5].

B

Discussion on Algorithm that Optimises for the Bound

An algorithm has been shown to find the target model that is guaranteed to exist according to theorem 3. [9] Its significance in using MSA theory for practical applications warrants a brief discussion of the optimisation problem solves by this algorithm.

Recall that [9]'s algorithm is not derived from the goal of the setting. Rather, it minimises the difference between the left- and right-hand sides of the bound in lemma 2:

$$\min_p L_{\mathcal{D}_k}(h(p)) - L_{\mathcal{D}_p}(h(p)) - \eta' \quad \forall k \tag{B.1}$$

Negative values of the objective satisfy the bound (eqn. 3.1) and by the theorem at least one exists. Positive values do not satisfy the bound. Thus the minimisation selects solutions that do satisfy the bound over those that do not and no explicit constraint is necessary.

It is equivalent to minimise not for all k but for the worst-case k, so an equivalent optimisation problem is:

$$\min_p \max_k L_{\mathcal{D}_k}(h(p)) - L_{\mathcal{D}_p}(h(p)) \tag{B.2}$$

Because η' is a constant so does not influence the optimum it is discarded. This optimisation problem is guaranteed to find a solution that satisfies the bound in theorem 2.

[9] proves its equivalence to another formulation of the objective and give an algorithm that solves it. Specifically, they prove this is a difference of convex decomposition (DC)-programming problem. The algorithm they give converges to a *local* optimum [9, sec. 4.3]. It is however possible to test for *global* optimality: if a solution is found that achieves $\gamma = 0$, that solution is globally optimal.

C

Out-of-Scope: Notes on Alternative TMAs and MSA Theory

Recall from section 3.2 that an alternative MSA setting with a different Target Mixture Assumption (TMA) would require adapting the proofs of lemma 2 and theorem 3 to yield similar MSA theory, i.e. with a similar bound. This is because in their proofs a crucial assumption is the TMA, which was used both indirectly in the fixed-point mapping (FPM) of lemma 2 and directly in theorem 3. A different FPM yields different MSA theory, both for the current MSA setting and for a different TMA.

In that light this appendix will review the role of the chosen FPM in the proofs of MSA theory. No definitive conclusion is drawn, but rather several notes follow that might be of interest for future work on MSA theory, in alternative as well as the current MSA setting.

C.1. A More General Fixed-Point Mapping

The following is a statement similar to lemma 2, but for more general Fixed-Point Mappings than the specific $[\Phi(z)]_k$ of equation A.1.

Lemma 8. *Consider a fixed-point mapping $\Phi(\theta) : \Theta \to \Theta$ of the form*

$$[\Phi(\theta)]_k = \mathit{normalise}\,(v_k(\cdot;\theta)) \tag{C.1}$$

with

$$v_k(\cdot;\theta) = \theta_k v'_k(\cdot;\theta) + v'_0(\cdot)$$

defined for $\Theta = [0, \theta_{max}]^K$ with $\theta_{max} < \infty$ and some functions $v'_k \geq 0$ and $v'_0 > 0$ that are respectively dependent on and independent of both k and θ. Also, v'_k is continuous in θ. (The normalisation is defined for some K-length vector x with $x_k \geq 0\ \forall k$ as $\mathit{normalise}\,(x_k) = \frac{x_k}{\sum_k x_k}$.)

Then there exists a $\theta \in \Theta, \theta_k \neq 0$ such that

$$v'_k(\cdot;\theta) \leq \sum_{j=1}^{K} v_j(\cdot;\theta) \quad \forall k \tag{C.2}$$

Proof. The fixed-point mapping is assumed of the form as in the statement of the lemma, i.e.

$$[\Phi(\theta)]_k = \text{normalise}\,(v_k(\cdot;\theta)) = \frac{\theta_k v'_k(\cdot;\theta) + v'_0(\cdot)}{\sum_{j=1}^{K} v_j(\cdot;\theta)} \tag{C.3}$$

Note that under the assumptions in the statement $v_k > 0$ so that the normalisation is always defined. Further note that v_k is continuous in θ so Φ is as well, and that Θ is a set such as required for theorem 7, making Φ a valid fixed-point mapping according to that theorem.

There then exists $\theta \in \Theta$ such that $\Phi(\theta) = \theta$ or equivalently for that θ and any k

$$\theta_k = \frac{\theta_k v'_k(\cdot;\theta) + v'_0(\cdot)}{\sum_{j=1}^{K} v_j(\cdot;\theta)}$$

where $\theta_k \neq 0$. (To see this, observe that since $v'_0 > 0$, $\theta_k = 0$ yields the contradiction $0 = \frac{1}{K} > 0$.) Rewriting yields

$$\begin{aligned} v'_k(\cdot;\theta) &= \sum_{j=1}^{K} v_j(\cdot;\theta) - \frac{v'_0(\cdot)}{\theta_k} \\ &\leq \sum_{j=1}^{K} v_j(\cdot;\theta) \end{aligned}$$

which is bounded by discarding the last term. This concludes the proof. □

C.2. On Designing a Fixed-Point Mapping

Other fixed-point mappings for use in MSA theory might be designed as follows. This example uses the current TMA.

Requirements for a mapping are:

- The TMA should be applicable, i.e.:
  - Parameter z should be a valid mixture parameter, i.e. $z \in \Delta$ and normalised $\sum_k z_k = 1$.
  - The mapping should include a term $\sum_k z_k L_{\mathcal{D}_k}(h)$ (so that the TMA can be applied).
- Brouwer's Fixed-Point Theorem should be applicable, i.e.:
  - Parameter z should be in a set that's compact, convex and non-empty.
  - Mapping Φ should be continuous in z.

To design Φ, first assume the unavoidable: $z \in \Delta$ and $\sum_k z_k = 1$. Applying Brouwer's means $z = \Phi(z)$ or equivalently $z_k = [\Phi(z)]_k \; \forall k$, which together with the normalisation implies $1 = \sum_k [\Phi(z)]_k$. Whatever $z = \phi(z)$ the mapping returns, it can be normalised as $[\Phi(z)]_k = \frac{[\phi(z)]_k}{\sum_j [\phi(z)]_j}$ to satisfy the condition. So now a $\phi(z)$ is desired so that Φ includes a term $\sum_k z_k L_{\mathcal{D}_k}(h)$ and is defined for and continuous in all $z \in \Delta$.

To end up with the desired term, $\phi = z_k L_{\mathcal{D}_k}(h)$ would be suitable. However, because it's possible that $\phi = 0$, after normalisation Φ might undefined.

One option, equivalent to the mapping used by [1, 8, 9], is to apply smoothing as $[\phi(z)]_k = z_k L_{\mathcal{D}_k}(h) + \eta'/K$ with $\eta' > 0$ needed for Φ to be defined for $z = 0$.

The cause of $\phi = 0$ is that equivalently either $z = 0$ and/or $L_{\mathcal{D}_k}(h) = 0$. An alternative smoothing might then target only one of $z = 0$ and $L_{\mathcal{D}_k}(h)$. Suppose the latter, so that $[\phi(z)]_k = z_k(L_{\mathcal{D}_k}(h) + \eta')$, leading to the alternative MSA theory of lemma 9.

To design a different mapping, note the following on the normalisation of Φ. The normalisation as done here does always work, but is only necessary if $\phi(z)$ is not yet normalised of itself. So firstly, for a designed mapping it can be checked if the numerator already evaluates to 1, always, and if so the normalisation can be removed.

Secondly, an alternative is to use a series of functions that are inherently normalised. Relevant keywords to investigate might be: partition of unity; Bernstein polynomials. Lastly, note that $+\eta'$ is not the only smoothing that prevents zeroes; one could for example apply the exponential function. However, all these suggestions lead to a completely different mapping that likely does not include the desired summation-term to apply the current TMA–but they might be applicable for a different TMA.

Lemma 9. *Consider K sources $\mathcal{D}_k$. There exists $z \in \Delta$ such that for any source k for which $z_k \neq 0$*

$$L_{\mathcal{D}_k}(h(z)) = \sum_{j=1}^{K} z_j L_{\mathcal{D}_j}(h(z)) \tag{C.4}$$

for a model $h(z)$ that is continuous in its parameter $z \in \Delta$.

Proof. Consider the fixed-point mapping $\Phi(z) : \Delta \to \Delta$ defined as

$$[\Phi(z)]_k = \text{normalise}\left(z_k \left(L_{\mathcal{D}_k}(h(z)) + \eta'\right)\right) = \frac{z_k \left(L_{\mathcal{D}_k}(h(z)) + \eta'\right)}{\sum_{j=1}^{K} z_j L_{\mathcal{D}_j}(h(z)) + \eta'} \tag{C.5}$$

where the normalisation is defined for some K-length vector x with $x_k \geq 0 \; \forall k$ as $\text{normalise}\,(x_k) = \frac{x_k}{\sum_k x_k}$ where $\sum_k x_k \neq 0$. The normalisation is always defined because η' guarantees the denominator is strictly positive.

Note that by assumption $L_{\mathcal{D}}(h(z))$ is continuous in z so Φ is as well, and that Δ is a set such as required for theorem 7, making Φ a valid fixed-point mapping according to that theorem.

There then exists a $z \in \Delta$ such that $\Phi(z) = z$ or equivalently for that z and any k

$$z_k = \frac{z_k \left(L_{\mathcal{D}_k}(h(z)) + \eta'\right)}{\sum_{j=1}^{K} z_j L_{\mathcal{D}_j}(h(z)) + \eta'}$$

This implies[1] that either $z_k = 0$, or if $z_k \neq 0$ we can rewrite this as

$$L_{\mathcal{D}_k}(h(z)) = \sum_{j=1}^{K} z_j L_{\mathcal{D}_j}(h(z))$$

This concludes the proof. □

[1]Note that $z_k = 0$ is not allowed in the proofs of [1, Lem. 2] and [9, Lem. 6].

Bibliography

[1] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. "Domain adaptation with multiple sources". In: *Advances in Neural Information Processing Systems 21 - Proceedings of the 2008 Conference* (2008), pp. 1041–1048.

[2] Peter H. Jin et al. *How to scale distributed deep learning?* 2016. arXiv: 1611.04581 [cs.LG].

[3] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern recognition*. 4th ed. Orlando, FL, USA: Academic Press, 2008. ISBN: 9781597492720.

[4] Gabriele Beate Schweikert et al. "An Empirical Analysis of Domain Adaptation Algorithms for Genomic Sequence Analysis". In: *NIPS*. Vol. 8. 2008, pp. 1433–1440.

[5] Mahesh Joshi et al. "Multi-domain learning: When do domains matter?" In: *EMNLP-CoNLL 2012 - 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Proceedings of the Conference* (2012), pp. 1302–1312.

[6] Zirui Wang et al. "Characterizing and Avoiding Negative Transfer". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 11285–11294. DOI: 10.1109/CVPR.2019.01155.

[7] Fangzhao Wu and Yongfeng Huang. "Sentiment domain adaptation with multiple sources". In: *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers* 1 (2016), pp. 301–310. DOI: 10.18653/v1/p16-1029.

[8] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. "Multiple source adaptation and the Rényi divergence". In: *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009* (2009), pp. 367–374.

[9] Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. "Algorithms and theory for multiple-source adaptation". In: *Advances in Neural Information Processing Systems* 2018-Decem.NeurIPS (2018), pp. 8246–8256. ISSN: 10495258. arXiv: 1805.08727.

[10] Shiliang Sun, Honglei Shi, and Yuanbin Wu. "A survey of multi-source domain adaptation". In: *Information Fusion* 24 (2015), pp. 84–92. ISSN: 15662535. DOI: 10.1016/j.inffus.2014.12.003.

[11] Shai Ben-David et al. "Analysis of representations for domain adaptation". In: *Advances in Neural Information Processing Systems* (2007), pp. 137–144. ISSN: 10495258. DOI: 10.7551/mitpress/7503.003.0022.

[12] John Shore and Rodney Johnson. "Properties of cross-entropy minimization". In: *IEEE Transactions on Information Theory* 27.4 (1981), pp. 472–482.

[13] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014. ISBN: 9781107057135.

[14] Muhammad Ghifary et al. "Scatter component analysis: A unified framework for domain adaptation and domain generalization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7 (2017), pp. 1414–1430. ISSN: 01628828. DOI: 10.1109/TPAMI.2016.2599532. arXiv: 1510.04373.