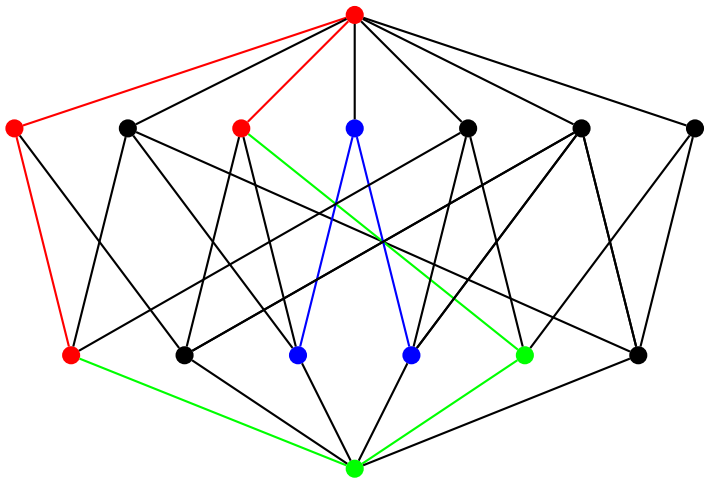
Meta-embeddings: a probabilistic generalization of embeddings in machine learning

Niko Brümmer[1] , Lukáš Burget, Paola Garcia,
Oldřich Plchot, Johan Rohdin, Daniel Romero,
David Snyder, Themos Stafylakis, Jesús Villalba

JHU HLTCOE 2017 SCALE Workshop



[1]Corresponding author: niko.brummer@gmail.com. The other authors are in alphabetical order.

Contents

List of Figures

Chapter 1

Introduction

Embeddings are familiar in modern machine learning. Neural nets to extract word embeddings[1] were already proposed in 2000 by Bengio [1]. Now embeddings are used more generally, for example in state-of-the-art face recognition, e.g. Facenet [2].

Embeddings are becoming popular also in speech and speaker recognition. In Interspeech 2017, eighteen papers had the word 'embedding' in the title.[2] In speaker recognition and spoken language recognition, we have been using i-vectors, a precursor to embeddings, for almost a decade [3, 4, 5]. More general embeddings are now appearing in speaker recognition, see for example the Voxceleb paper [6] and David Snyder's work [7, 8]. A new embedding method for language recognition is proposed in [9].

Input patterns (sequences of acoustic feature vectors, images, text, ...) live in large, complex spaces, where probability distributions and geometrical concepts such as distance are difficult to formulate. Embeddings extracted from the input patterns live in simpler spaces, e.g. $\mathbb{R}^d$ (multidimensional Euclidean space), where distance is naturally defined and can be put to work to compare patterns.

At the Johns Hopkins HLTCOE SCALE 2017 Workshop[3] one of the topics of interest was embeddings for speaker recognition. During the workshop, the idea to generalize to *meta-embeddings* was conceived. At the time of the writing of this document, we are still developing the theory and very little coding and experiments have been done.[4] This document describes the theory.

[1] en.wikipedia.org/wiki/Word_embedding

[2] www.interspeech2017.org/program/technical-program/.

[3] http://hltcoe.jhu.edu/research/scale/scale-2017

[4] This document and some code to produce some of the examples in it can be found at https://github.com/bsxfan/meta-embeddings.

We envisage that meta-embeddings could be widely applicable to a variety of machine learning or pattern recognition problems, but we shall make our discussion concrete by using the language of automatic speaker verification/recognition. To interpret everything (say) as face recognition, just do *recording* $\mapsto$ *image* and *speaker* $\mapsto$ *face*, and so on ....

1.1 Meta-embeddings

An important concept used throughout this document is that of the *hidden identity variable*—a hypothetical, multidimensional variable that ideally contains all of the information that distinguishes one speaker/face/individual from others. In speaker recognition, the hidden *speaker identity variable* is well known from the work of Patrick Kenny in JFA [10] and PLDA [11]. Similar identity variables appeared in earlier face recognition applications of PLDA [12, 13, 14].

We argue that the way embeddings are currently treated, essentially makes them *point estimates of hidden variables*. We propose instead to let the identity variables—and therefore the embeddings—remain hidden and to instead extract *meta-embeddings*, which are probabilistic representations of what the values of the hidden variables might be. For example, if the embedding lives in $\mathbb{R}^d$, then the meta-embedding could be in the form of a multivariate normal distribution, where the mean could act as point estimate, but where there is also a covariance matrix that quantifies the uncertainty around the mean.

Quantifying the uncertainty is very important if the recognizer is to be applicable to variable and sometimes challenging conditions. In speaker recognition, a short, noisy, narrow-band recording should leave much more uncertainty about the speaker than a long, clean, wideband recording. In face recognition, compare a well-lit, high resolution, full-frontal face image to a grainy, low resolution, partly occluded face. In fingerprint recognition, compare a clean, high-resolution ten-print, to a single, distorted, smudged fingermark retrieved from a crime scene.

The idea to represent the uncertainty is not novel. Indeed, it is *obvious* that to do things properly, we must take the uncertainty into account. The problem is that it turns out to be difficult to do in practice.

- Part of the problem is computational. For example, a point estimate for an identity variable in $\mathbb{R}^d$ can be represented as a d-dimensional vector. But for a multivariate normal meta-embedding, the covariance is a d-by-d matrix, which is more expensive to store and manipulate.

- Then there is the added complication that covariance matrices have to be positive definite. If we use a neural net to compute our uncertainty for us, we have to specially structure the neural net to respect this requirement.

- But the problem has another facet—if a neural net manufactures meta-embeddings for us, how do we know that the uncertainty thus represented is reasonable? Given meta-embeddings extracted from some supervised database, what criterion can we apply to measure the goodness of the information and the uncertainty they contain? Such criteria are important, because we need them for discriminative training of our neural nets.

This document will provide some suggestions for tackling the above problems.

The main purpose of the document is however to show that the meta-embeddings are themselves embeddings in the sense that they live in a vector space, equipped with geometric and algebraic structure that can be employed to compute probabilistic recognition scores (likelihood ratios). The hope is that the theoretical part of this document will help to provide the structure to guide further research into finding better solutions for the practical problems of working with this kind of uncertainty.

1.2 Prior work

In [15], Gaussian embeddings are proposed to represent uncertainty. To be expanded, ...

Mention uncertainty propagation in i-vectors[16, 17, 18, 19], ...

Chapter 2

Distillation of meta-embeddings from independence assumptions

In this chapter we use probability theory and some basic independence assumptions to motivate and derive the concept of meta-embeddings.

We use speaker recognition terminology, but everything is applicable more widely (images, etc. ...). Our inputs are *voice recordings*, each assumed to contain a single speaker. Recordings can have various representations, i-vectors, sequences of feature vectors, the raw speech samples, etc. .... At the writing of this document, recordings represented as feature vector sequences are of primary interest.

In general, we will be interested in *sets* of recordings, ranging from the set of all the recordings in a training database, down to just a pair of recordings in a verification trial. We represent a set of n recordings as:

$$\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$$

We shall work with hypotheses about how such sets of recordings are partitioned w.r.t. speaker and we shall make use of independence assumptions conditioned on these hypotheses.

2.1 Independence assumptions

We base everything that follows on a pair of simple independence assumptions:

- Multiple recordings of the same speaker are *exchangeable*.
- Recordings of different speakers are *independent*.

Almost all current probabilistic solutions in speaker recognition—also PLDA—make use of these assumptions. While these assumptions are mathematically convenient, in the real world they are not exactly true:

- The speech of any speaker will change over a time span of years and recordings made over such a period will not be exchangeable.
- The independence assumption between different speakers can only apply if our recognizer is sufficiently well trained: The recognizer has to have been exposed to so much speech, that only the speech of a new speaker itself can provide information about that speaker.

As long as we understand the limitations imposed by our assumptions, we can proceed.

In what follows, we use the notation H_1 for the hypothesis that the set of recordings of interest, $\mathcal{R} = \{r_1, \ldots, r_n\}$, were all spoken by the same (one) speaker. For the hypothesis that they were spoken by m different speakers, we use H_m. Exchangeability means:

$$P(\mathcal{R} \mid H_1) = P(r_1, r_2 \ldots, r_n \mid H_1) = P(r_2, r_1, \ldots, r_n \mid H_1) = \cdots$$

where the joint probability does not depend on the order of the recordings. According to De Finetti's exchangeability theorem,[1] the only way to obtain exchangeability is to introduce some hidden variable, say $\mathbf{z} \in \mathcal{Z}$, such that the recordings are *conditionally independent* given $\mathbf{z}$. Marginalizing over $\mathbf{z}$ w.r.t. some prior distribution $\pi(\mathbf{z})$, gives the exchangeable joint distribution [20]:

$$P(\mathcal{R} \mid H_1) = \Big\langle \prod_{i=1}^{n} P(r_i \mid \mathbf{z}) \Big\rangle_{\mathbf{z} \sim \pi} \tag{2.1}$$

where the angle brackets denote expectation—for continuous $\mathbf{z}$, this could be written as an integral and for discrete $\mathbf{z}$ as a summation. By this marginalization, we are inducing dependence between recordings of the same speaker, while retaining exchangeability. *We do need this dependence, otherwise speaker recognition would be impossible!*

Let $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m$ be a number of non-overlapping sets of recordings, spoken respectively by m different speakers. The between-speaker independence

[1] en.wikipedia.org/wiki/Exchangeable_random_variables

assumption gives:

$$P(\mathcal{R}_1, \ldots, \mathcal{R}_m \mid m \text{ speakers}) = \prod_{i=1}^{m} P(\mathcal{R}_i \mid H_1) = \prod_{i=1}^{m} \Big\langle \prod_{r \in \mathcal{R}_i} P(r \mid \mathbf{z}) \Big\rangle_{\mathbf{z} \sim \pi} \tag{2.2}$$

where we have expanded $P(\mathcal{R}_i \mid H_1)$ using (2.1).

According to this model, the hidden variable, $\mathbf{z} \in \mathcal{Z}$, contains everything that is to be known about recordings of a speaker. Patrick Kenny calls $\mathbf{z}$ the *speaker identity variable*. At this stage, we are not committing ourselves to the nature of the hidden variables and therefore we leave $\mathcal{Z}$ undefined—after all, $\mathbf{z}$ is hidden so there is much freedom in choosing its nature. (If you need something concrete to shape your thoughts, $\mathcal{Z} = \mathbb{R}^d$ is a good choice.)

Think of the identity variable, $\mathbf{z}$, as the ideal embedding that an oracle could extract. If we were given $\mathbf{z}$, the problem would be instantly solved. Unfortunately, there is noise, distortion, occlusion and all kinds of other mechanisms that induce uncertainty, so we can never exactly extract $\mathbf{z}$. Let's not pretend that we can do this. Let us not extract some point-estimate for $\mathbf{z}$ and call that our embedding. Let's instead extract a *meta-embedding*, which is *information about* the ideal embedding, $\mathbf{z}$, where the information has a probabilistic form. We derive this form below.

2.2 Speaker identity likelihood functions

Given a pair of recordings, $\mathcal{R} = \{r, r'\}$, we can answer the question of whether they belong to the same speaker or not in terms of the *likelihood-ratio (LR)* [21]:

$$\begin{aligned} \frac{P(r, r' \mid H_1)}{P(r, r' \mid H_2)} &= \frac{\big\langle P(r \mid \mathbf{z})\ P(r' \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}}{\big\langle P(r \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi} \big\langle P(r' \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}} \\ &= \frac{\big\langle kP(r \mid \mathbf{z})\ k'P(r' \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}}{\big\langle kP(r \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi} \big\langle k'P(r' \mid \mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}} \\ &= \frac{\big\langle f_r(\mathbf{z})\ f_{r'}(\mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}}{\big\langle f_r(\mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi} \big\langle f_{r'}(\mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi}} \end{aligned} \tag{2.3}$$

where we have defined the *speaker identity likelihood functions*:

$$f_r(\mathbf{z}) = kP(r \mid \mathbf{z}) \qquad \text{and} \qquad f_{r'}(\mathbf{z}) = k'P(r' \mid \mathbf{z}), \tag{2.4}$$

where $k, k' > 0$ are arbitrary constants that may depend on r, r', but not on $\mathbf{z}$. Notice that the LR depends on the data (r, r') only via f_r and $f_{r'}$. The speaker information in r is represented by the *whole function*

$$f_r : \mathcal{Z} \to \mathbb{R}$$

rather than by the function value $f_r(\mathbf{z})$ at some particular value of $\mathbf{z}$ (remember $\mathbf{z}$ is hidden and we are never given a fixed value for it).

The full speaker information cannot be represented by some estimate of $\mathbf{z}$. We could for example use $\hat{\mathbf{z}} = \operatorname{argmax} f_r(\mathbf{z})$ as a maximum-likelihood point-estimate for $\mathbf{z}$, but that would be throwing information away. The full information about the speaker is contained in the whole function, f_r.

The representation of the speaker information can be further understood by noticing that f_r and $f_{r'}$ represent r, r' in any posteriors for the speaker identity variable, for example:

$$P(\mathbf{z} \mid r, \pi) = \frac{\pi(\mathbf{z}) f_r(\mathbf{z})}{\left\langle f_r(\mathbf{z}') \right\rangle_{\mathbf{z}' \sim \pi}} \tag{2.5}$$

and if r and r' are known to be of the same speaker, then:

$$P(\mathbf{z} \mid r, r', H_1, \pi) = \frac{\pi(\mathbf{z}) f_r(\mathbf{z}) f_{r'}(\mathbf{z})}{\left\langle f_r(\mathbf{z}') f_{r'}(\mathbf{z}') \right\rangle_{\mathbf{z}' \sim \pi}} \tag{2.6}$$

2.3 The general case

Our speaker identity likelihood functions are applicable more generally than just to pairs of recordings. Under the independence assumptions of section 2.1, speaker identity likelihood functions can be used to answer *any* question that can be formulated in terms of partitioning a set of recordings according to speaker [22]. Let $\mathcal{R} = \{r_1, \ldots, r_n\}$ be a set of recordings and let $A : \mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m \subseteq \{1, 2, \ldots, n\}$ be a hypothesized partition of $\mathcal{R}$ into subsets belonging to $m \geq 1$ different hypothesized speakers.[2] Similarly, let $B : \mathcal{S}'_1, \ldots, \mathcal{S}'_{m'}$ be a different hypothesized partition, having m' hypothesized speakers. A and B are labels that refer to the two alternate partitioning hypotheses. Using within-speaker exchangeability and between-speaker

[2] The subsets are non-empty, non-overlapping and their union equals $\{1, 2, \ldots, n\}$. We allow $m = 1$, in which case $\mathcal{S}_1 = \mathcal{R}$.

independence, the LR comparing A to B is:

$$\begin{aligned}\frac{P(\mathcal{R} \mid A)}{P(\mathcal{R} \mid B)} &= \frac{\prod_{i=1}^{m} \left\langle \prod_{j \in \mathcal{S}_i} P(r_j \mid \mathbf{z}) \right\rangle_{\mathbf{z} \sim \pi}}{\prod_{i=1}^{m'} \left\langle \prod_{j \in \mathcal{S}'_i} P(r_j \mid \mathbf{z}) \right\rangle_{\mathbf{z} \sim \pi}} \\ &= \frac{\prod_{i=1}^{m} \left\langle \prod_{j \in \mathcal{S}_i} f_j(\mathbf{z}) \right\rangle_{\mathbf{z} \sim \pi}}{\prod_{i=1}^{m'} \left\langle \prod_{j \in \mathcal{S}'_i} f_j(\mathbf{z}) \right\rangle_{\mathbf{z} \sim \pi}}\end{aligned} \tag{2.7}$$

where, as above, the *speaker identity likelihood function* extracted from r_j is:

$$f_j(\mathbf{z}) = k_j P(r_j \mid \mathbf{z}) \tag{2.8}$$

The last equality in (2.7) follows because the arbitrary scaling constants, $\{k_j\}_{j=1}^{n}$, are the same in the numerator and denominator and cancel. This LR form is convenient because of this cancellation.

Any kind of speaker recognition problem that can be expressed in terms of partitioning recordings according to speaker, can be scored at runtime via likelihood-ratios of the form (2.7). We shall show some examples below. Moreover, any discriminative training criteria that are expressed as functions of the LR scores can then also be expressed in terms of (2.7). In summary:

> *For discriminative speaker recognition, all scoring and training calculations can be expressed in terms of the speaker identity likelihood functions.*

To better appreciate this generality, we shall present some examples in the next section.

Again, for readers that are comfortable with the idea that probability distributions convey information [23], the speaker identity likelihood functions are closely associated with posteriors for $\mathbf{z}$. Any such posterior, conditioned on a number of recordings of a speaker indexed by $\mathcal{S}$, can be computed as:

$$P(\mathbf{z} \mid \mathcal{R}, \mathcal{S}, H_1, \pi) = \frac{\pi(\mathbf{z}) \prod_{j \in \mathcal{S}} f_j(\mathbf{z})}{\left\langle \prod_{j \in \mathcal{S}} f_j(\mathbf{z}') \right\rangle_{\mathbf{z}' \sim \pi}} \tag{2.9}$$

The likelihood function, f_j, or indeed a product of any number of them, can be regarded as a *raw* form of posterior distribution for $\mathbf{z}$. We use the term raw, because the likelihood function is independent of the choice of prior, π, and is also un-normalized. Nevertheless, f_j contains all the speaker information that a given probabilistic model could extract from r_j.

2.4 Examples

The examples below should aid understanding of the meaning and applications of (2.7). We will frequently refer back to these examples in the rest of this document.

Simple verification. Let $\mathcal{R} = \{r_1, r_2\}$, $A : \mathcal{S}_1 = \{1, 2\}$ and $B : \mathcal{S}'_1 = \{1\}, \mathcal{S}'_2 = \{2\}$. Then $\frac{P(\mathcal{R}|A)}{P(\mathcal{R}|B)}$ is the canonical (single-enroll) speaker verification LR.

Multi-enroll verification. Let $\mathcal{R} = \{r_1, r_2, r_3\}$, $A : \mathcal{S}_1 = \{1, 2, 3\}$ and $B : \mathcal{S}'_1 = \{1, 2\}, \mathcal{S}'_2 = \{3\}$. Then $\frac{P(\mathcal{R}|A)}{P(\mathcal{R}|B)}$ is the speaker verification LR, with enrollment recordings r_1, r_2 and test recording r_3.

Open-set classification. Let $\mathcal{R} = \{r_1, \ldots, r_n\}$ be partitioned into m speakers, indexed by $\mathcal{S}_1, \ldots, \mathcal{S}_m \subseteq \{1, \ldots, n\}$. Let $r' \neq \mathcal{R}$ be an additional (test) recording. For $1 \leq i \leq m$, let A_i be the hypothesis that r' is spoken by speaker i, as indexed by $\mathcal{S}_i$. Let A_{m+1} be the hypothesis that there are $m+1$ speakers: m of them indexed by $\mathcal{S}_1, \ldots, \mathcal{S}_m$ and r' spoken by a new speaker. Then we can score the open-set speaker classification problem as follows: For $1 \leq i \leq m+1$, the likelihood[3] for A_i is

$$L_i = \frac{P(\mathcal{R}, r' \mid A_i)}{P(\mathcal{R}, r' \mid A_{m+1})} \tag{2.10}$$

Notice that likelihood that r' was spoken by a new speaker is just $L_{m+1} = 1$. Assuming some hypothesis prior, $P(A_i)$, the hypothesis posterior is:

$$P(A_i \mid \mathcal{R}, r') = \frac{P(A_i)L_i}{\sum_{j=1}^{m+1} P(A_j)L_j} \tag{2.11}$$

Agglomerative hierarchical clustering. Let $\mathcal{R} = \{r_1, \ldots, r_n\}$ be partitioned into m hypothesized speakers, indexed by $A : \mathcal{S}_1, \ldots, \mathcal{S}_m \subseteq \{1, \ldots, n\}$. For $1 \leq i, j \leq m$ and $i \neq j$, let B_{ij} be the hypothesis that the speakers are correctly indexed as given by A, except that the recordings indexed by $\mathcal{S}_i$ and $\mathcal{S}_j$ are from the same speaker. That is, if we accept hypothesis B_{ij}, then

[3]The numerator of L_i by itself, $P(\mathcal{R}, r' \mid A_i)$, could also serve as likelihood for A_i. Since the denominator is independent of i, the normalized ratio, L_i, is still a valid likelihood for A_i, with the advantage that the troublesome data-dependent constants cancel.

we should reduce the number of speakers to $m-1$ and merge $\mathcal{S}_i$ and $\mathcal{S}_j$. We let

$$L_{ij} = \frac{P(\mathcal{R} \mid B_{ij})}{P(\mathcal{R} \mid A)} \tag{2.12}$$

be the (conveniently normalized) likelihood for hypothesis B_{ij}, while the likelihood for hypothesis A is just 1. These likelihood scores can be used as part of an iterative, greedy optimization[4] procedure (agglomerative hierarchical clustering) to find a partition of $\mathcal{R}$ according to speaker, with high posterior probability.

Summary. All of these examples are scored using likelihood ratios of the form (2.7) and are therefore dependent on the data only via the speaker identity likelihood functions.

We have so far demonstrated that f_j qualifies as an embedding in the sense that it contains all of the relevant information in r_j. In what follows we shall refer to $\mathbf{z}$ as the (hidden, ideal) *embedding*, while the f_j are our *meta-embeddings*.

It remains to be shown in the next chapter that these meta-embeddings live in a space with useful algebraic and geometric structure.

[4]Finding the global optimum is apparently hopelessly intractable. For a set of $n = 75$ recordings, we already have that the number of possible ways to partition this set exceeds the number of atoms (about 10^{80}) in the observable universe.

Chapter 3

The structure of meta-embedding space

Classical embeddings in machine learning live in finite-dimensional, Euclidean space, $\mathbb{R}^d$. When equipped with the standard scalar product, vector addition, dot-product and Euclidean distance, $\mathbb{R}^d$ is a vector space, inner-product space, metric space, Hilbert space, etc. Our meta-embeddings, the functions f_j, live in a slightly generalized space, which is still a vector space, inner product space, metric space and (subject to regularity conditions) a Hilbert space.

Depending on the nature of the hidden variable, $\mathbf{z} \in \mathcal{Z}$, our meta-embeddings might live in a space with very high, or even infinite dimensionality. In the theory that follows, everything can be more elegantly described in this high/infinite-dimensional space, but in practice—to be discussed in the next chapter—we will consider various finite, tractable representations of our meta-embeddings.

In this chapter we explore first the geometric (vector space) properties of meta embeddings that establish concepts such as length, distance and angle. The vector space operations include addition, scalar multiplication and inner product.

Second, we enrich the vector space by introducing another (elementwise) multiplicative operator, which then forms an algebra, the properties of which we explore. This paves the way for the next chapter where we assemble these algebraic manipulations to compute likelihood-ratios of the form (2.7).

3.1 Geometric properties

Let $\mathbb{R}^{\mathcal{Z}}$ denote the space of all possible functions from $\mathcal{Z}$ to $\mathbb{R}$ and let our meta-embeddings live in $\mathcal{F}$, where $\mathcal{F} \subset \mathbb{R}^{\mathcal{Z}}$. We need to confine meta-embeddings to some subset of $\mathbb{R}^{\mathcal{Z}}$ in order to exclude all kinds of pathological functions and to ensure that the operations defined on meta-embeddings are well behaved. Since our meta-embeddings are likelihoods, the function values are always non-negative, so we might consider restricting ourselves to $\mathbb{R}_+^{\mathcal{Z}}$, where $\mathbb{R}_+ = [0, \infty)$. But this does not give a vector space, because it will not be closed w.r.t. scalar multiplication. It will therefore be more convenient to work in a more general space, $\mathcal{F}$, that includes functions with negative values.

3.1.1 Functions as vectors

Consider the Euclidean vector, $\mathbf{x} \in \mathbb{R}^d$ and notice that $\mathbf{x}$ can be viewed as a *function* that maps the component index to the real line: $\{1, \ldots, d\} \to \mathbb{R}$.

Now let the hidden speaker identity variable be a d-dimensional Bernoulli vector, $\mathbf{z} \in \mathcal{Z} = \{0, 1\}^d$. That is, $\mathbf{z}$ is a vector with d components, each of which can be either 0 or 1. A function, $f : \{0, 1\}^d \to \mathbb{R}$, can be represented as a vector having 2^d real components. More generally, whenever $\mathcal{Z}$ has finite cardinality, $f : \mathcal{Z} \to \mathbb{R}$ can be represented as a vector in Euclidean space.

However, for the continuous case, when $\mathbf{z} \in \mathbb{R}^d$, the function $f(\mathbf{z})$ is like a vector with an infinite number of components. For the vector-space operations of *scalar multiplication* and *addition*, the infinite dimensionality poses no difficulty. These operations are defined as follows. Let $f, g, h \in \mathcal{F}$ and $\alpha, \beta \in \mathbb{R}$, then:

$$h = \alpha f + \beta g \qquad \Leftrightarrow \qquad h(\mathbf{z}) = \alpha f(\mathbf{z}) + \beta g(\mathbf{z}) \tag{3.1}$$

As long as we choose $\mathcal{F}$ such that it is closed under these operations and to include the constant function with value 0, then $\mathcal{F}$ is a vector space, even though the vectors are infinite-dimensional. To quality as a vector space, the properties required of the addition and scalar multiplication operators — commutativity, associativity and distributivity—follow directly from (3.1).

3.1.2 Inner product

We already have that $\mathcal{F}$ is a vector space. To further make it an *inner product space*, we need to define the inner product, which is a function, $\mathcal{F} \times \mathcal{F} \to \mathbb{R}$. For $f, g, h \in \mathcal{F}$, and $\alpha, \beta \in \mathbb{R}$, we define our inner product as:

$$\big\langle f, g \big\rangle = \big\langle f(\mathbf{z}) g(\mathbf{z}) \big\rangle_{\mathbf{z} \sim \pi} \tag{3.2}$$

which is the expectation of the product of f and g, w.r.t π, a probability distribution over $\mathcal{Z}$. If we choose π to be our hidden variable prior, then the inner product coincides with the numerator in the likelihood-ratio of (2.3). Notice that we use the same triangular bracket notation for both inner product and expectation—this ambiguous notation will be convenient later.

For (3.2) to be a valid inner product, it needs to satisfy symmetry, linearity, and positive definiteness. The symmetry, $\langle f, g \rangle = \langle g, f \rangle$ and linearity, $\langle \alpha f + \beta g, h \rangle = \alpha \langle f, h \rangle + \beta \langle g, h \rangle$, follow directly from the definition. It also follows that $\langle f, f \rangle \geq 0$, which is necessary for positive definiteness. However, we also need that $\langle f, f \rangle = 0$ if and only if $f(\mathbf{z}) = 0$ everywhere. For this we need $\pi(\mathbf{z})$ to be non-zero everywhere on $\mathcal{Z}$ and we need $\mathcal{F}$ to impose certain smoothness constraints—e.g. we need to exclude from $\mathcal{F}$ functions that differ from each other only on subsets of $\mathcal{Z}$ of measure zero. In what follows, we shall assume that $\mathcal{F}$ is chosen such that it forms a valid inner-product space together with the operations defined here.[1]

Notice that (3.2) is a generalization of the usual dot product in Euclidean space. It generalizes the simple summation in the dot product by introducing the weighting by $\pi(\mathbf{z})$. For continuous $\mathcal{Z}$, summation is further generalized to integration.

Is $\mathcal{F}$ a Hilbert space? A Hilbert space is an inner product space with the additional requirement of *completeness*. If $\mathcal{F}$ is chosen to have this property, then yes, we have a Hilbert space. In what follows, we will not make use of completeness and we will therefore refer to our space with (slightly) greater generality as an inner product space.

3.1.3 Norm, distance, angle

The inner product naturally supplies the notions of length, distance and angle, making our space a metric space and a normed space. The *norm, or length* is defined as:

$$\|f\| = \sqrt{\langle f, f \rangle} \tag{3.3}$$

Once we have length, *distance* is defined as the length of the difference between two vectors, so that squared distance is given by:

$$\|f - g\|^2 = \langle f - g, f - g \rangle = \|f\|^2 + \|g\|^2 - 2\langle f, g \rangle \tag{3.4}$$

[1] An alternative construction of the inner product space could be to use the quotient space, populated by equivalence classes of functions, where two functions, $f(\mathbf{z})$ and $g(\mathbf{z})$ are equivalent if $P(f(\mathbf{z}) = g(\mathbf{z}) \mid \mathbf{z} \sim \pi) = 1$. See [24].

Conversely, it may be useful in some situations to express inner product in terms of norms. We can solve for $\langle f, g \rangle$ in (3.4), but notice that we can alternatively use:

$$\|f + g\|^2 = \langle f + g, f + g \rangle = \|f\|^2 + \|g\|^2 + 2\langle f, g \rangle \tag{3.5}$$

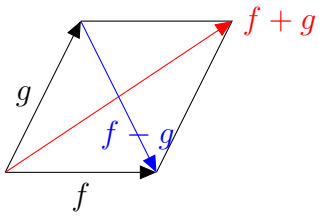
Addition of (3.4) and (3.5) gives the *parallelogram law*:

$$\|f + g\|^2 + \|f - g\|^2 = 2\|f\|^2 + 2\|g\|^2 \tag{3.6}$$

Combining everything, we can express the *inner product in terms of norms* in at least three different ways:

$$\begin{aligned} \langle f, g \rangle &= \frac{1}{2}\left(\|f\|^2 + \|g\|^2 - \|f - g\|^2\right) \\ &= \frac{1}{2}\left(\|f + g\|^2 - \|f\|^2 - \|g\|^2\right) \\ &= \frac{1}{4}\left(\|f + g\|^2 - \|f - g\|^2\right) \end{aligned} \tag{3.7}$$

To better understand this geometry, view the parallelogram, with sides f, g, and diagonals $f + g$ and $f - g$:



Finally, we can also define angle. For any inner product space we have the *Cauchy-Schwartz inequality*:

$$\left|\langle f, g \rangle\right| \leq \|f\| \, \|g\| \tag{3.8}$$

so that the *angle* between f and g, say θ, can be defined as:

$$\cos\theta = \frac{\langle f, g \rangle}{\|f\| \, \|g\|} \tag{3.9}$$

because then $|\cos\theta| \leq 1$, as it should.

Having defined angle, we can also talk about pairs of orthogonal vectors, for which the inner product is zero and which have an angle of $\pi/2$ between them. For orthogonal vectors, with $\langle f, g \rangle = 0$, we can use (3.4) to derive the *theorem of Pythagoras*:

$$\|f - g\|^2 = \|f\|^2 + \|g\|^2 \tag{3.10}$$

3.1.4 Note on meta-embeddings as random variables

This note is of theoretical interest and is not essential to understanding the rest of the document.

Let $(\mathcal{Z}, \mathcal{A}, \pi)$ be a probability space, where $z \in \mathcal{Z}$ is the identity variable as defined above. $\mathcal{A}$ is a suitable sigma-algebra, consisting of subsets of $\mathcal{Z}$. The probability measure of this space, $\pi : \mathcal{A} \to [0, 1]$ is the prior on z. Under the regularity condition that our meta-embeddings are measurable functions from $\mathcal{Z}$ to $\mathbb{R}$, the meta-embeddings can be viewed as *random variables* [25]. Informally, when f_j is a meta-embedding and if we sample $z \sim \pi$, then the likelihood $f_j(z)$ is a sample from the random variable f_j.

The interested reader is encouraged to read both [24] and [26] where it is explained how the expectation of the product of two random variables, i.e. our (3.2), forms a well-defined inner product. Further reading may include appendix B of [27] and references therein.

3.2 Algebraic properties

Having described the inner-product space, we now enrich our meta-embedding space with the elementwise multiplicative operation. This operation combines in useful ways with the other operations to enable us to do the calculations we need for our ultimate goal of computing the likelihood ratios of the form (2.7).

Alert readers may notice that our addition operator does not seem to play any useful, practical role in all of this. First, the fact that addition between meta-embeddings is possible helps to establish that we have a vector space. Once this is established, all of the well-known properties of vector spaces become available to us. (For example, to derive the basic property of distance, we need subtraction, which in turn is defined by scalar multiplication and addition.) Moreover, in the next chapter, we do find a practical application for addition, in the form of mixture distributions.

3.2.1 Elementwise product

We now equip our space with the *elementwise product* (Hadamard product). Since this product is associative, we can write it using juxtaposition, defining it as:

$$h = fg \qquad \Leftrightarrow \qquad h(\mathbf{z}) = f(\mathbf{z})g(\mathbf{z}) \tag{3.11}$$

The vector space, $\mathcal{F}$, equipped additionally with the elementwise product forms an *algebra*, because the elementwise product is bilinear w.r.t. addition

and scalar multiplication.[2]

We introduce the elementwise product to represent *pooling* of the relevant information extracted from multiple recordings. Let $\mathcal{R} = \{r_1, \ldots, r_n\}$ be a set of recordings of the same speaker. If the corresponding meta-embeddings are $f_j(\mathbf{z}) = k_j P(r_j \mid \mathbf{z})$, then

$$f_{\mathcal{R}}(\mathbf{z}) = \prod_{j=1}^{n} f_j(\mathbf{z}) = \big(\prod_j k_j\big) P(\mathcal{R} \mid \mathbf{z}) \tag{3.12}$$

gives the *pooled meta-embedding*, $f_{\mathcal{R}} = \prod_j f_j$, which represents all of the speaker information in $\mathcal{R}$, in the sense that:

$$P(\mathbf{z} \mid \mathcal{R}, H_1, \pi) = \frac{\pi(\mathbf{z}) f_{\mathcal{R}}(\mathbf{z})}{\big\langle f_{\mathcal{R}}(\mathbf{z}')\big\rangle_{\mathbf{z}'\sim\pi}} \tag{3.13}$$

3.2.2 Multiplicative identity

We define the *multiplicative identity* as the constant function with value 1 everywhere: $\mathbf{1}(\mathbf{z}) = 1$. This turns out to be a (conveniently normalized) meta-embedding extracted from any *empty recording*. An empty recording is any recording that contains no speech—it could have zero duration, or otherwise contain silence, noise, or any other non-speech sounds. Since empty recordings have no speaker information, speaker identity variable posteriors conditioned on them are just equal to the prior. Letting r_ϕ be an empty recording, we have:

$$P(\mathbf{z} \mid r_\phi, \pi) = \frac{\mathbf{1}(\mathbf{z})\pi(\mathbf{z})}{\big\langle \mathbf{1}(\mathbf{z})\big\rangle_{\mathbf{z}\sim\pi}} = \pi(\mathbf{z}) \tag{3.14}$$

Notice that:

$$\|\mathbf{1}\| = \sqrt{\big\langle 1^2\big\rangle_{\mathbf{z}\sim\pi}} = 1 \tag{3.15}$$

so that $\mathbf{1}$ represents the *diagonal axis* of length 1, with all 'components' equal to 1. The inner product, $\big\langle f, \mathbf{1}\big\rangle$, will be of special interest below. It is the length of the orthogonal projection of f onto this diagonal axis—see section 4.4 for a graphical example.

[2] Compare this to the complex numbers, which under addition and (real) scalar multiplication, form a 2-dimensional vector space. If complex multiplication is added, it is also an algebra. See: https://en.wikipedia.org/wiki/Algebra_over_a_field.

3.2.3 L1 and L2 norms

We have already introduced the norm, $\|f\| = \sqrt{\langle f, f \rangle}$. Observe that this is an L2 norm:

$$\|f\| = \sqrt{\langle f, f \rangle} = \sqrt{\langle f^2(\mathbf{z}) \rangle_{\mathbf{z} \sim \pi}} = \|f\|_2 \tag{3.16}$$

We shall also find the L1 norm useful. Notice that when f is non-negative, we can interpret $\langle f, \mathbf{1} \rangle$ as an L1 norm:

$$\|f\|_1 = \langle |f(\mathbf{z})| \rangle_{\mathbf{z} \sim \pi} \tag{3.17}$$

It is easy to show using (3.15) and Cauchy-Schwartz (3.8) that:

$$\|f\|_1 \leq \|f\|_2 \tag{3.18}$$

If f is non-negative, then $\langle f, 1 \rangle = \|f\|_1 \leq \|f\|_2$. More generally:

$$|\langle f, 1 \rangle| \leq \|f\|_2 \tag{3.19}$$

Below we restrict attention to meta-embeddings for which $|\langle f, \mathbf{1} \rangle| < \infty$ and it is useful to know that this condition is automatically satisfied if $\|f\| = \|f\|_2 < \infty$.

3.2.4 Normalized meta-embedding

Let $f \in \mathcal{F}$ be a meta-embedding. We define the corresponding *normalized meta-embedding* as:

$$\overline{f} = \frac{1}{\langle f, \mathbf{1} \rangle} f \qquad \Leftrightarrow \qquad \overline{f}(\mathbf{z}) = \frac{f(\mathbf{z})}{\langle f(\mathbf{z}') \rangle_{\mathbf{z}' \sim \pi}} \tag{3.20}$$

If f_r is the meta-embedding for recording r and $f_r(\mathbf{z}) = kP(r \mid \mathbf{z})$, then:

$$P(\mathbf{z} \mid r, \pi) = \pi(\mathbf{z}) \overline{f_r}(\mathbf{z}) \tag{3.21}$$

Similarly, for a set of recordings, $\mathcal{R}$, all of the same speaker, let $f_{\mathcal{R}}$ be the pooled meta-embedding, such that $f_{\mathcal{R}}(\mathbf{z}) = kP(\mathcal{R} \mid \mathbf{z}, H_1)$, then we can rewrite (3.13) more compactly as:

$$P(\mathbf{z} \mid \mathcal{R}, H_1, \pi) = \pi(\mathbf{z}) \overline{f_{\mathcal{R}}}(\mathbf{z}) \tag{3.22}$$

We shall require that all meta-embeddings are *normalizable* in the sense that $|\langle f, 1 \rangle| < \infty$. By (3.19), $\|f\| < \infty$ is a sufficient condition. Alternatively, any bounded function, $f(\mathbf{z})$ also satisfies this requirement.

Properties

The following properties of normalized meta-embeddings are easily shown. By definition, we have that the length of the projection of any normalized meta-embedding onto $\mathbf{1}$ is unity:

$$\langle \overline{f}, \mathbf{1} \rangle = 1 \tag{3.23}$$

This gives that *differences between normalized meta-embeddings* live in a hyperplane perpendicular to $\mathbf{1}$:

$$\langle \overline{f} - \overline{g}, \mathbf{1} \rangle = 0 \tag{3.24}$$

By (3.18) we have:

$$\left\| \overline{f} \right\| \geq 1 \tag{3.25}$$

with equality at $\overline{f} = \mathbf{1}$. Using Pythagoras (3.10), we find the interesting formula:

$$\left\| \overline{f} \right\|^2 = \|\mathbf{1}\|^2 + \left\| \overline{f} - \mathbf{1} \right\|^2 = 1 + \left\| \overline{f} - \mathbf{1} \right\|^2 \tag{3.26}$$

The sum of two normalized meta-embeddings is easy to renormalize:

$$\overline{\overline{f} + \overline{g}} = \frac{1}{2}(\overline{f} + \overline{g}) \tag{3.27}$$

which, when used with (3.26), gives:

$$\left\| \frac{1}{2}(\overline{f} + \overline{g}) \right\|^2 = \left\| \overline{\overline{f} + \overline{g}} \right\|^2 = 1 + \left\| \frac{1}{2}\overline{f} + \frac{1}{2}\overline{g} - \mathbf{1} \right\|^2 \tag{3.28}$$

or

$$\left\| \overline{f} + \overline{g} \right\|^2 = 4 + \left\| (\overline{f} - \mathbf{1}) + (\overline{g} - \mathbf{1}) \right\|^2 \tag{3.29}$$

This compares to:

$$\left\| \overline{f} - \overline{g} \right\|^2 = \left\| (\overline{f} - \mathbf{1}) - (\overline{g} - \mathbf{1}) \right\|^2 \tag{3.30}$$

which inspires the definition of *centered* meta-embeddings:

3.2.5 Centered meta-embeddings

For a meta-embedding f, the normalized meta-embedding is $\overline{f}$ and the *centered* meta embedding is:

$$\tilde{f} = \overline{f} - \mathbf{1} \tag{3.31}$$

The inner product is:

$$\begin{aligned} \langle \tilde{f}, \tilde{g} \rangle &= \langle \overline{f} - \mathbf{1}, \overline{g} - \mathbf{1} \rangle \\ &= \langle \overline{f}, \overline{g} \rangle - \langle \overline{f}, \mathbf{1} \rangle - \langle \mathbf{1}, \overline{g} \rangle + \langle \mathbf{1}, \mathbf{1} \rangle \\ &= \langle \overline{f}, \overline{g} \rangle - 1 \end{aligned} \tag{3.32}$$

By (3.7), we therefore have:

$$\begin{aligned} \langle \overline{f}, \overline{g} \rangle &= 1 + \langle \tilde{f}, \tilde{g} \rangle \\ &= 1 + \frac{1}{2}\left(\left\|\tilde{f}\right\|^2 + \|\tilde{g}\|^2 - \left\|\tilde{f} - \tilde{g}\right\|^2\right) \\ &= 1 + \frac{1}{2}\left(\left\|\tilde{f} + \tilde{g}\right\|^2 - \left\|\tilde{f}\right\|^2 - \|\tilde{g}\|^2\right) \\ &= 1 + \frac{1}{4}\left(\left\|\tilde{f} + \tilde{g}\right\|^2 - \left\|\tilde{f} - \tilde{g}\right\|^2\right) \end{aligned} \tag{3.33}$$

3.2.6 Notations for inner product and expectation

Using the definitions of inner product and elementwise multiplication, notice that $\langle f, g \rangle = \langle fg, \mathbf{1} \rangle$ and indeed that $\langle fgh, \mathbf{1} \rangle = \langle fg, h \rangle = \langle f, gh \rangle = \langle \mathbf{1}, fgh \rangle$, and so on. The comma in the inner product notation has no real function and it can be omitted. We shall write:

$$\langle f, g \rangle = \langle fg \rangle = \langle f(\mathbf{z})g(\mathbf{z}) \rangle_{\mathbf{z}\sim\pi} \tag{3.34}$$

and more generally,

$$\langle \prod_j f_j \rangle = \langle \prod_j f_j(\mathbf{z}) \rangle_{\mathbf{z}\sim\pi} \tag{3.35}$$

In what follows, we shall interchangeably use the notations with or without comma, to respectively emphasize the dot-product or expectation interpretations.

Chapter 4

Likelihood-ratios

We are now ready to demonstrate the utility of our meta-embeddings operators. We show how they can be used to express the LR formulas of chapter 2.

We start with *simple LRs*, which can be used to compare any two hypotheses for the partitioning of a set of recordings according to speaker, where the partitions differ by splitting or combining a single subset. Then we treat the general case, where we compute LRs that compare arbitrary hypotheses. The chapter concludes with a graphical toy example to aid understanding.

4.1 Simple LRs

Here we show how to write the LR of each of the examples of section 2.4 as an inner product between normalized meta-embeddings.

The *simple verification* likelihood ratio (2.3), rewritten in a variety of notations, is:

$$\frac{P(r, r' \mid H_1)}{P(r, r' \mid H_2)} = \frac{\langle f, f' \rangle}{\langle f, \mathbf{1} \rangle \langle f', \mathbf{1} \rangle} = \frac{\langle f f' \rangle}{\langle f \rangle \langle f' \rangle} = \langle \overline{f}, \overline{f'} \rangle \tag{4.1}$$

where f and f' are the meta-embeddings extracted from r and r'.

The *multi-enroll verification* LR, with enrollments: $r_1 \mapsto f_1$ and $r_2 \mapsto f_2$ and test: $r_3 \mapsto f_3$, is:

$$\frac{P(r_1, r_2, r_3 \mid H_1)}{P(r_1, r_2, r_3 \mid H_2)} = \frac{\langle f_1 f_2, f_3 \rangle}{\langle f_1 f_2, \mathbf{1} \rangle \langle f_3, \mathbf{1} \rangle} = \frac{\langle f_1 f_2 f_3 \rangle}{\langle f_1 f_2 \rangle \langle f_3 \rangle} = \langle \overline{f_1 f_2}, \overline{f_3} \rangle \tag{4.2}$$

For *open-set classification*, with enrollment recordings, $\mathcal{R} = \{r_1, \ldots, r_n\}$ for m speakers, indexed by $\mathcal{S}_1, \ldots, \mathcal{S}_m \subseteq \{1, \ldots, n\}$ and a new test recording, r', we use the meta-embeddings: $\{r_\ell \mapsto f_\ell\}_{\ell=1}^n$ and $r' \mapsto f'$ and we calculate the LR that r' is of speaker i, vs that r' is a new speaker, as:

$$\begin{aligned} L_i &= \frac{\langle f' \prod_{\ell \in \mathcal{S}_i} f_\ell \rangle \prod_{j \neq i} \langle \prod_{\ell \in \mathcal{S}_j} f_\ell \rangle}{\langle f' \rangle \prod_{j=1}^m \langle \prod_{\ell \in \mathcal{S}_j} f_\ell \rangle} \\ &= \frac{\langle f' \prod_{\ell \in \mathcal{S}_i} f_\ell \rangle}{\langle f' \rangle \langle \prod_{\ell \in \mathcal{S}_i} f_\ell \rangle} \\ &= \langle \overline{f'}, \overline{\prod_{\ell \in \mathcal{S}_i} f_\ell} \rangle \end{aligned} \tag{4.3}$$

For *agglomerative clustering* we can similarly write the LR (2.12) in terms of normalized meta-embeddings as:

$$L_{ij} = \frac{\langle \prod_{\ell \in \mathcal{S}_i \cup \mathcal{S}_j} f_\ell \rangle}{\langle \prod_{\ell \in \mathcal{S}_i} f_\ell \rangle \langle \prod_{\ell \in \mathcal{S}_i} f_\ell \rangle} = \langle \overline{\prod_{\ell \in \mathcal{S}_i} f_\ell}, \overline{\prod_{\ell \in \mathcal{S}_j} f_\ell} \rangle \tag{4.4}$$

In summary: In each of these simple cases, the likelihood-ratio is most compactly represented as an *inner product between normalized meta-embeddings*. In what follows, we refer to these inner products more concisely as *normalized inner products*.

4.2 The general case

If we consider the most general case of LRs of the form (2.7), we find that there are cases that cannot be written as a single normalized inner product. But we show that they *can* however be written as products and ratios of such inner products. We start with an example.

Example 1. Consider a set of four recordings, $\mathcal{R} = \{r_1, r_2, r_3, r_4\}$ with associated meta-embeddings, $\{f_1, f_2, f_3, f_4\}$ and consider the two hypotheses:

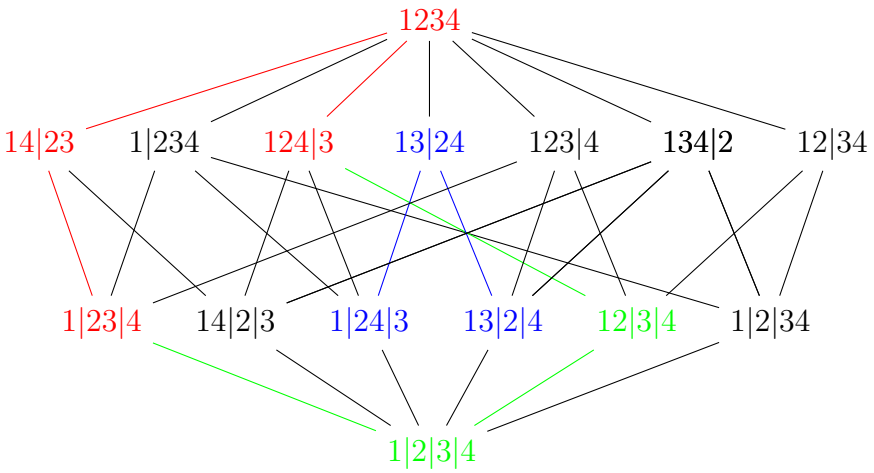


Figure 4.1: Hasse diagram of the lattice of partitions of a set of 4 recordings, ordered by 'refines'. Every downward arc is an an atomic refinement, which splits a single subset. Example 1 is highlighted in blue, example 2a in red and 2b in green.

$A : 1|24|3$ and $B : 13|2|4$, written in a convenient short-hand. The LR is:

$$\begin{aligned}
\frac{P(\mathcal{R} \mid A)}{P(\mathcal{R} \mid B)} &= \frac{\langle f_1 \rangle \langle f_2 f_4 \rangle \langle f_3 \rangle}{\langle f_1 f_3 \rangle \langle f_2 \rangle \langle f_4 \rangle} \\
&= \frac{\langle f_1 \rangle \langle f_2 f_4 \rangle \langle f_3 \rangle}{\langle f_1 f_3 \rangle \langle f_2 f_4 \rangle} \times \frac{\langle f_1 f_3 \rangle \langle f_2 f_4 \rangle}{\langle f_1 f_3 \rangle \langle f_2 \rangle \langle f_4 \rangle} \\
&= \frac{\langle f_1 \rangle \langle f_3 \rangle}{\langle f_1 f_3 \rangle} \times \frac{\langle f_2 f_4 \rangle}{\langle f_2 \rangle \langle f_4 \rangle} \\
&= \frac{\langle \overline{f_2}, \overline{f_4} \rangle}{\langle \overline{f_1}, \overline{f_3} \rangle}
\end{aligned} \tag{4.5}$$

We have achieved this re-arrangement by observing that we can get from A to B via the auxiliary hypothesis $C : 13|24$, where C is reached from A by joining $\{1\}\{3\}$ and B is reached from C by splitting $\{2, 4\}$. The join contributes a normalized inner product factor below the line and the split contributes another above the line.

In general, the partitions of a set form a *partial order*, where $A < C$ is defined as: *C is finer than A*. In our example, we have both $A < C$ and $B < C$, but A and B are *not directly comparable*. If two hypotheses *are* directly comparable (if they differ by joining or splitting a single subset), then their LR can be expressed as a single normalized inner product, as we

have seen in section 4.1.

More can be said about this partial order. It is in fact a *lattice*, where every pair of partitions has a unique supremum (least upper bound) and a unique infimum (greatest lower bound). Figure 4.1 shows the Hasse diagram representing the lattice of four recordings. Every arc in the Hasse diagram corresponds to splitting or joining a single subset. As we shall see, there is a simple LR associated with every arc.

To form the LR between any pair of hypotheses, we can traverse the lattice from one hypothesis to the other via any path connected by arcs, but the shortest paths will go via the supremum or the infimum. Since the infimum and supremum always exist, there will always be a path. Every step (split or join)[1] contributes a normalized inner product factor to the final LR.

In example 1 we traversed the blue path via $13|24 = \sup(A, B)$. If we instead traverse via $1|2|3|4 = \inf(A, B)$, we get the *same* decomposition. This is however not true in general—different paths can give different decompositions of the LR. This is shown in the next example.

Example 2a. Let's try a more complex path, highlighted in red in figure 4.1. Let $A : 1|23|4$ and $B : 124|3$. We have $\sup(A, B) = 1234$ and $\inf(A, B) = 1|2|3|4$. Whether we go up or down, we need at least three steps to traverse between A and B. There are three such paths via the supremum and three more via the infimum. Following the red highlighted path, the LR can be re-arranged as:

$$\begin{aligned}\frac{P(\mathcal{R} \mid A)}{P(\mathcal{R} \mid B)} &= \frac{\langle f_1\rangle\langle f_2 f_3\rangle\langle f_4\rangle}{\langle f_1 f_2 f_4\rangle\langle f_3\rangle} \\ &= \frac{\langle f_1\rangle\langle f_2 f_3\rangle\langle f_4\rangle}{\langle f_1 f_4\rangle\langle f_2 f_3\rangle} \times \frac{\langle f_1 f_4\rangle\langle f_2 f_3\rangle}{\langle f_1 f_2 f_3 f_4\rangle} \times \frac{\langle f_1 f_2 f_3 f_4\rangle}{\langle f_1 f_2 f_4\rangle\langle f_3\rangle} \\ &= \frac{1}{\langle \overline{f_1}, \overline{f_4}\rangle} \times \frac{1}{\langle \overline{f_1 f_4}, \overline{f_2 f_3}\rangle} \times \langle \overline{f_1 f_2 f_4}, \overline{f_3}\rangle\end{aligned} \tag{4.6}$$

Each upward step (join) contributes a normalized inner product below the line, while the downward step (split) contributes another above.

Example 2b. Let's also try a path via the infimum, highlighted in green, to traverse between the same two nodes as before. Now the LR can be

[1] In lattice theory, infimum and supremum are alternatively termed *meet* and *join*. In our usage here, we mean by *join* simply to form the union of two subsets of recordings.

re-arranged as:

$$\begin{aligned}\frac{P(\mathcal{R} \mid A)}{P(\mathcal{R} \mid B)} &= \frac{\langle f_1 \rangle \langle f_2 f_3 \rangle \langle f_4 \rangle}{\langle f_1 f_2 f_4 \rangle \langle f_3 \rangle} \\ &= \frac{\langle f_1 \rangle \langle f_2 f_3 \rangle \langle f_4 \rangle}{\langle f_1 \rangle \langle f_2 \rangle \langle f_3 \rangle \langle f_4 \rangle} \times \frac{\langle f_1 \rangle \langle f_2 \rangle \langle f_3 \rangle \langle f_4 \rangle}{\langle f_1 f_2 \rangle \langle f_3 \rangle \langle f_4 \rangle} \times \frac{\langle f_1 f_2 \rangle \langle f_3 \rangle \langle f_4 \rangle}{\langle f_1 f_2 f_4 \rangle \langle f_3 \rangle} \\ &= \langle \overline{f_2}, \overline{f_3} \rangle \times \frac{1}{\langle \overline{f_1}, \overline{f_2} \rangle} \times \frac{1}{\langle \overline{f_1 f_2}, \overline{f_4} \rangle}\end{aligned} \tag{4.7}$$

Again, each upward step (join) contributes a normalized inner product below the line, while the downward step (split) contributes another above. This rule applies in general. If we follow a path from the numerator (hypothesis A), to the denominator (hypothesis B), joins contribute below the line, while splits contribute above. (If we go from denominator to numerator, this rule is reversed.)

Equivalence of paths

As the two paths that we examined in example 2 demonstrate, there may be many different paths and therefore many different decompositions of a given LR in terms of normalized inner products. Nevertheless, as long as our calculations are exact, all such decompositions must give the same numerical result. This may serve as a useful regularization constraint if we are training neural nets to approximate inner products between meta-embeddings. (As we shall see in the next chapter, there may be various practical reasons to prefer approximate calculations.)

The equivalence of paths is not limited to shortest paths. The products and ratios of the normalized inner products of any path between two nodes must give the same numerical result, irrespective of the path. This is also true in particular of *cycles*. If we start and end at the same node, the LR we calculate thus is *unity*. The interesting thing about this is that if we are given a set of unlabelled recordings, we can use this fact to manufacture a (very) large set of constraints on the values of normalized inner products. (If we work with logarithms, this becomes a homogeneous system of linear equations.) These constraints may perhaps help to regularize neural nets in both labelled and unlabelled training regimes.

Primitive operations

Which primitive building blocks do we need to compute general LRs of the form (2.7)? Let's assume we can always extract raw (unnormalized) meta-

embeddings from any recording. Then, as (2.7) shows, we can compute any LR if we can:

- arbitrarily pool meta-embeddings (do elementwise multiplication) and
- compute arbitrary expectations of single or pooled meta-embeddings.

Alternatively, as we have shown in this section, we can compute arbitrary LRs if we can

- pool and
- compute arbitrary normalized inner products.

We can however somewhat restrict the requirement on normalized inner products. We can get away with inner products where *one of the arguments is always a single (unpooled) meta-embedding*. To do normalization, we need $\langle f, \mathbf{1} \rangle$, which respects this restriction. To further enforce this restriction, look again at the lattice in figure 4.1 and convince yourself that you can traverse between any two nodes by always adding or removing but a single element of some subset in the partition. For example, the red path does not respect this restriction, but the blue and green ones do. For such paths, all the inner product factors will have at least one unpooled argument.

When we discuss practical representations of meta-embeddings in the next chapter, we will see that for some of the representations, pooling may change the form of the representation. The form may have implications on how easy it is to do inner products. If one of the arguments is always unpooled, this may facilitate calculations.

Finally, we mention another interesting building block. It is not difficult to show that we can form arbitrary LRs from factors of the form:

$$\frac{\langle \prod_{i=1}^{m} f_i \rangle}{\prod_{i=1}^{m} \langle f_i \rangle} = \left\langle \prod_{i=1}^{m} \overline{f_i} \right\rangle \tag{4.8}$$

for $m = 1, 2, \ldots$. This factor is just the LR comparing the coarsest to the finest partition of a set of m recordings. It can be argued that this building block is not primitive, because it can be assembled via pooling and expectation (or pooling and inner products).

4.3 Note on cosine similarity

It is of interest to note that our normalized inner product (LR):

$$\langle \overline{f_1}, \overline{f_2} \rangle = \frac{\langle f_1, f_2 \rangle}{\|f_1\|_1 \, \|f_2\|_1}$$

is very similar to the popular cosine similarity between traditional embeddings, say $\mathbf{x}_1, \mathbf{x}_2$, given by:

$$\frac{\langle \mathbf{x}_1, \mathbf{x}_2 \rangle}{\|\mathbf{x}_1\|_2 \, \|\mathbf{x}_2\|_2}$$

The difference is the use of L1 vs L2 normalization.

In the literature, there are many examples where embeddings are L2-normalized before computing dot products or distances. In speaker recognition, the first i-vector scoring recipe used cosine-similarity [3], and indeed in current PLDA scoring recipes, it is still standard practice to use L2-normalized i-vectors [28]. In face recognition, Facenet uses cosine similarity to compare embeddings [2], although subsequently [29] advises against such normalization. In [9], the embeddings for language recognition are compared with cosine similarity.

Notice that by (3.4), if embeddings have unit L2 norm, then there is no essential difference (other than the sign) between cosine similarity (inner product) and squared distance. By the Cauchy-Schwartz inequality (3.8), inner products between L2-normalized embeddings are limited to the range $[-1, 1]$, where -1 corresponds to embeddings on opposite sides of the unit hypersphere (maximum distance) and 1 to embeddings that coincide (zero distance).

In speaker recognition, cosine similarity between embeddings in Euclidean space is regarded as an uncalibrated form of *log* likelihood-ratio. (The cosine similarity is signed, just like the log likelihood-ratio.) In contrast, our inner products between L1-normalized meta-embeddings give non-negative *likelihood-ratios*. Notice also that by (3.18), our likelihood-ratio magnitudes are *not* limited to be at most unity.

4.4 Toy example, with binary hidden variable

Let us now construct a graphical example to reinforce our geometric understanding of our meta-embeddings. In order to be able to plot the meta-embeddings, we choose perhaps the simplest possible meta-embeddings, which live in $\mathbb{R}^2$.

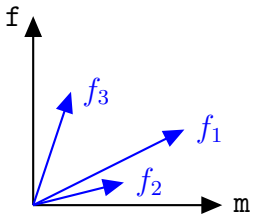
We do this by imagining a very weak speaker recognizer (meta-embedding extractor) that can only distinguish voices by whether they sound male or female. That is, we choose the simplest possible hidden speaker identity variable by making it discrete and binary: $\mathbf{z} \in \{\mathtt{m}, \mathtt{f}\}$. We choose the prior to have males and females equally likely:

$$\pi(\mathtt{m}) = \pi(\mathtt{f}) = \frac{1}{2}$$

Seen as a function, the meta-embedding for recording r_i is $f_i(\mathbf{z}) = k_i P(r_i \mid \mathbf{z})$. Since $\mathbf{z}$ has only two possible values, we can represent the meta-embedding as a *two-dimensional vector*: $f_i = \big[k_i P(r_i \mid \mathtt{m}), k_i P(r_i \mid \mathtt{f})\big]$. Because of the prior weighting, $\pi(\mathtt{m}) = \pi(\mathtt{f}) = \frac{1}{2}$, the inner product and dot product differ by a factor 2: if $f_i = [f_{i1}, f_{i2}]$, then:

$$\big\langle f_i, f_j \big\rangle = \frac{1}{2}(f_{i1}f_{j1} + f_{i2}f_{j2}) \tag{4.9}$$

For three hypothetical recordings, let the meta-embeddings be $f_1 = [2, 1]$, $f_2 = [1.2, 0.3]$, $f_3 = [0.5, 1.5]$ and we can plot these meta-embeddings as:



The first thing to notice is the difference between embedding and meta-embedding: The ideal embedding here is binary, $\mathbf{z} \in \{\mathtt{m}, \mathtt{f}\}$, while the meta-embedding is continuous: $f_i \in \mathbb{R}^2$. Here, and in general, the meta-embedding lives in a more complex space than $\mathbf{z}$.

Since likelihoods are positive, the meta-embeddings are confined to the positive quadrant. We should never be working outside of this quadrant, but to see the space as a vector space, we need to be aware of the existence of the other quadrants. Indeed, when we design our neural nets to extract meta-embeddings, we will have to make sure they end up being in the positive quadrant.

Because of the arbitrary scaling constants, k_i, the lengths of our meta-embedding vectors do not carry information—but from the directions we see that f_1 and f_2 are probably male, while f_3 is probably female. Our end-goal is not to infer the genders of the speakers, but to infer whether the speakers of different recordings are the same or not. A little thought should convince the reader that:

- The smallest possible likelihood-ratio, $\frac{P(r_i, r_j|H_1)}{P(r_i, r_j|H_2)} = 0$, would be obtained if we were certain that one speaker is male and the other female. This would happen when we have:

  $$P(r_i \mid \mathtt{m}) \gg P(r_i \mid \mathtt{f}) \qquad \text{and} \qquad P(r_j \mid \mathtt{m}) \ll P(r_j \mid \mathtt{f})$$

  so that f_i is on the horizontal (male) axis, while f_j is on the vertical (female) axis. Then indeed, the inner product (and dot product)

between these two orthogonal meta-embeddings would be zero and so would the LR.

- The largest possible LR with the weak, binary hidden variable would be just 2, which would be obtained when we are certain that both recordings are of the same gender. We need to consider details of the normalization to see how this works out.

Figure 4.2 shows the same three meta-embeddings, together with their normalized versions, with normalization as defined by (3.20). Remember the prior-weighting: when $f_i = [f_{i1}, f_{i2}]$, then the normalization constant is

$$\langle f_i, \mathbf{1} \rangle = \frac{1}{2}(f_{i1} + f_{i2})$$

Inner products between the normalized meta-embeddings give the likelihood-ratios,

$$\frac{P(r_i, r_j \mid H_1)}{P(r_i, r_j \mid H_2)} = \langle \overline{f_i}, \overline{f_j} \rangle$$

Since the discrimination given by the binary hidden variable is weak, the likelihood-ratios are not too far from the neutral value of 1. The LR, $\langle \overline{f_1}, \overline{f_2} \rangle = 1.2$ is greater than 1, slightly favouring the hypothesis that the speakers of r_1 and r_2 are the same. The other two LRs are less than one, favouring the H_2 hypothesis in each case. The LR, $\langle \overline{f_2}, \overline{f_3} \rangle = 0.64$ is *stronger* (further from 1) than $\langle \overline{f_1}, \overline{f_3} \rangle = 0.8$, because we are more certain of the maleness of f_2 than we are of f_1. The largest possible LR of 2 would be obtained when both normalized meta-embeddings coincide on the horizontal or vertical axis, at $[2, 0]$, or $[0, 2]$.[2]

Figure 4.3 shows what happens when we know that r_1 and r_2 are of the same speaker and we pool their meta-embeddings, giving us more certainty that this is a male speaker and more certainty that this speaker is different from the (probably female) speaker of r_3.

[2]Remember the prior weighting of $\frac{1}{2}$.

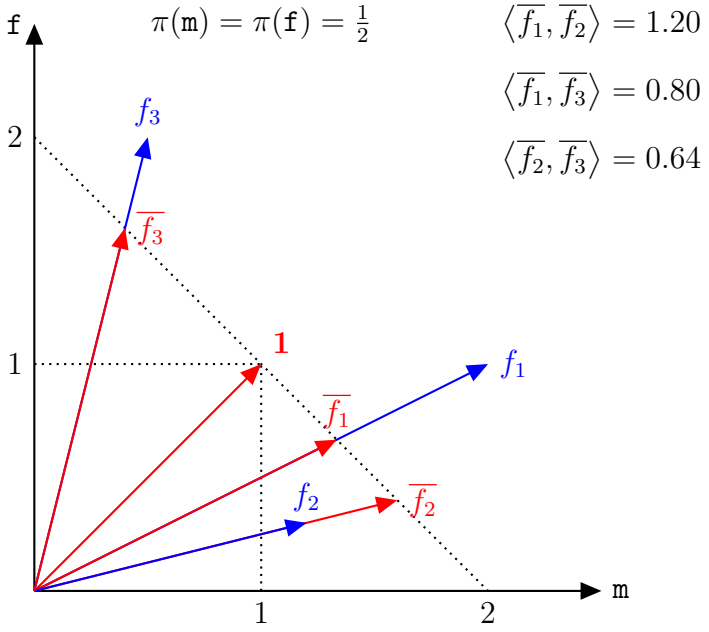


Figure 4.2: Normalized meta-embeddings (red) and their inner products. Raw meta-embeddings (blue) are normalized by scaling them so that their projections on **1** are unity.

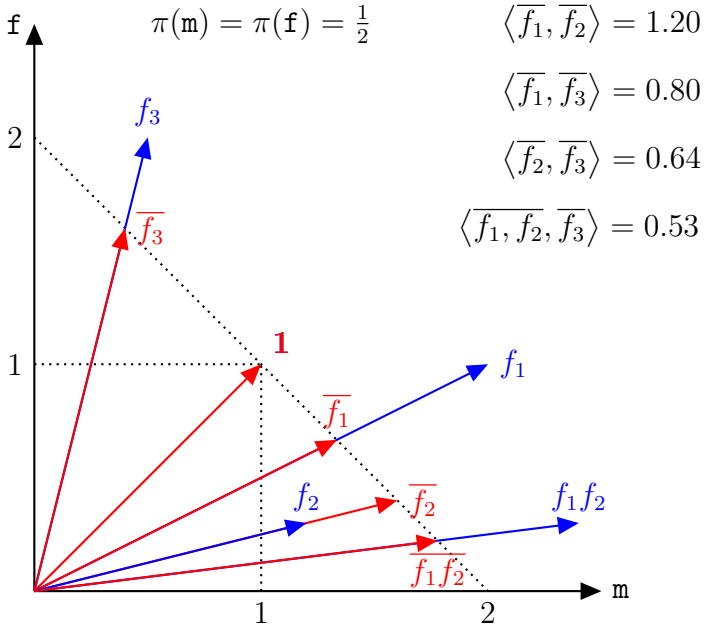


Figure 4.3: Pooling: If we know that f_1 and f_2 are from the same speaker, we can pool them, using the elementwise product $f_1 f_2$. This increases the certainty that this is a male speaker. Correspondingly, the strength (difference from 1) of $\langle \overline{f_1 f_2}, \overline{f_3} \rangle$ is more than either of $\langle \overline{f_1}, \overline{f_3} \rangle$ and $\langle \overline{f_2}, \overline{f_3} \rangle$. Pooling has increased the certainty that the speaker represented by $f_1 f_2$ is not the same speaker as the probably female f_3.

Chapter 5

Practical meta-embeddings

We have thus far developed a theoretical idea of the nature of meta-embeddings. We are now ready to explore a few proposals of how to practically represent meta-embeddings. The end-goal is to train a neural net that takes voice recordings as input and outputs representations for the corresponding meta-embeddings. We shall discuss training criteria for the neural net later. Here we are interested in the form that practical meta-embedding representations might take and we shall explore several possibilities.

Before proceeding to these possibilities, let us consider in general, some desirable properties of our representations. Ideally, we need all of the following properties for our meta-embeddings:

Non-negativity: $f(\mathbf{z}) \geq 0$ everywhere.

Normalizability: $\langle f, \mathbf{1} \rangle < \infty$.

Pooling should be tractable, and the pooled result, $f_i f_j$, should have the same representation as f_i and f_j.

Expectation: $\langle f(\mathbf{z}) \rangle_{\mathbf{z} \sim \pi}$ should be tractable for any meta-embedding f, whether it is raw or pooled.

Backpropagation of derivatives through pooling and expectation is necessary for training.

Recall that if we can pool and do expectations, then we can also do any inner products, since $\langle f, g \rangle = \langle fg \rangle$.

The first two proposals below, based on exponential family distributions, meet all of these requirements, although the expectations are somewhat computationally expensive. In some of the other proposals, various approximations and compromises have to be made.

5.1 GME: Gaussian meta-embedding

An obvious candidate for a practical meta-embedding is the multivariate Gaussian. A big advantage of this representation is that everything can be computed in closed form and that makes it an important building block in everything that follows. We shall refer to *Gaussian meta-embeddings* with the acronym GME.

This representation supposes a continuous speaker identity variable, $\mathbf{z} \in \mathbb{R}^d$. A trainable neural net processes each recording, r_j, and outputs the corresponding meta-embedding, f_j, in the form of a $(d + D)$-dimensional representation:

$$r_j \mapsto \mathbf{e}_j = (\mathbf{a}_j, \mathbf{b}_j)$$

where $\mathbf{a} \in \mathbb{R}^d$ and $\mathbf{b}_j = (b_{1j}, \ldots, b_{Dj}) \in \mathbb{R}^D_+$, where $D \geq 0$ and the $b_{ij} \geq 0$. (We allow the degenerate case, $D = 0$ to include also PLDA, where meta-embedding precisions are constant.) While the representation, $\mathbf{e}_j$, is finite-dimensional, it parametrizes the *infinite-dimensional* meta-embedding, f_j, defined as:

$$f_j(\mathbf{z}) = \exp\big[\mathbf{a}_j'\mathbf{z} - \frac{1}{2}\mathbf{z}'\mathbf{B}_j\mathbf{z}\big] \tag{5.1}$$

where $\mathbf{B}_j$ is a d-by-d, positive semi-definite precision matrix, composed as a conical combination:

$$\mathbf{B}_j = \mathbf{E}_0 + \sum_{i=1}^{D} b_{ij}\mathbf{E}_i \tag{5.2}$$

where $\{\mathbf{E}_i\}_{i=0}^D$ are fixed, d-by-d, positive semi-definite definite matrices—they are fixed in the sense of being independent of the input data (recordings), but the elements of these matrices are still trainable, together with the parameters of the neural net that extracts the $\mathbf{e}_j$. These matrices can be full, low rank, diagonal, etc.

Since $\mathbf{z} \in \mathbb{R}^d$ is hidden, we are free to choose the dimensionality, d. Experience with PLDA suggests $100 \leq d \leq 200$ is a good choice. This can also be compared to the Facenet embeddings, which are 120-dimensional. The size, D, of the precision parametrization can be used to trade off computational complexity vs capacity. To keep the complexity significantly less than that which would be needed for fully specified precision matrices, we probably want to constrain $D \ll \frac{d(d+1)}{2}$.

Scalar multiplication could be easily done in this framework but in practice, we probably won't need it. Addition of the meta-embeddings would give mixtures of Gaussians, with more complex representations, but for this representation we don't need addition.

5.1.1 Elementwise product

The important elementwise product is done by simply adding the representation vectors. For $\mathcal{R} = \{r_1, \ldots, r_n\}$, we can extract the individual representations, $\{\mathbf{e}_j\}_{j=1}^n$, which respectively represent the $\{f_j\}_{j=1}^n$. The representation for $f_{\mathcal{R}} = \prod_{j=1}^n f_j$ is then $\mathbf{e}_{\mathcal{R}} = \sum_{j=1}^n \mathbf{e}_j$. The representation $\mathbf{e}_j$ is essentially logarithmic, which gives us the benefit of automatic positivity, as well as easy elementwise multiplication. The disadvantage is somewhat complex expectation computation.

Note on group structure

For future reference, it might be useful to note that since Gaussians are closed under (associative) multiplication, our space of Gaussian meta-embeddings is a *monoid* (a semigroup, with identity). Since we exclude precision matrices with negative eigenvalues, our monoid lacks inverse elements, which would have made it a full group. The identity element is a Gaussian with zero mean and zero precision, which is just our previously defined identity, $\mathbf{1}(\mathbf{z}) = 1$.

5.1.2 Prior

To do expectations we need to define the prior. The simplest choice is the standard Gaussian:[1]

$$\pi(\mathbf{z}) = \mathcal{N}(\mathbf{z} \mid \mathbf{0}, \mathbf{I}) = \frac{e^{-\frac{1}{2}\mathbf{z}'\mathbf{z}}}{\sqrt{(2\pi)^d}} \tag{5.3}$$

5.1.3 Expectation (L1 norm)

Since our logarithmic representation is closed under elementwise multiplication, our expectations can always be performed on representations of the form (5.1). Dropping the subscript j to avoid clutter, we derive an expression

[1] Note here and elsewhere, we use two different fonts to differentiate $\pi(\mathbf{z})$, from the trigonometric constant π.

for $E(\mathbf{a}, \mathbf{B}) = \big\langle f \big\rangle$, where $f(\mathbf{z}) = \exp[\mathbf{a}'\mathbf{z} - \frac{1}{2}\mathbf{z}'\mathbf{B}\mathbf{z}]$:

$$\begin{aligned} E(\mathbf{a}, \mathbf{B}) &= \big\langle f \big\rangle \\ &= \int_{\mathbb{R}^d} f(\mathbf{z})\pi(\mathbf{z})\, d\mathbf{z} \\ &= \int_{\mathbb{R}^d} f(\mathbf{z})\mathcal{N}(\mathbf{z} \mid \mathbf{0}, \mathbf{I})\, d\mathbf{z} \\ &= \int_{\mathbb{R}^d} \frac{\exp\big[\mathbf{a}'\mathbf{z} - \frac{1}{2}\mathbf{z}'(\mathbf{I}+\mathbf{B})\mathbf{z}\big]}{\sqrt{(2\pi)^d}}\, d\mathbf{z} \end{aligned} \tag{5.4}$$

If we define $\boldsymbol{\mu} = (\mathbf{I} + \mathbf{B})^{-1}\mathbf{a}$, we can rewrite this as:

$$\begin{aligned} E(\mathbf{a}, \mathbf{B}) &= \big\langle f \big\rangle \\ &= \int_{\mathbb{R}^d} \frac{\exp\big[\boldsymbol{\mu}'(\mathbf{I}+\mathbf{B})\mathbf{z} - \frac{1}{2}\mathbf{z}'(\mathbf{I}+\mathbf{B})\mathbf{z}\big]}{\sqrt{(2\pi)^d}}\, d\mathbf{z} \\ &= \frac{\exp\big[\frac{1}{2}\boldsymbol{\mu}'(\mathbf{I}+\mathbf{B})\boldsymbol{\mu}\big]}{|\mathbf{I}+\mathbf{B}|^{\frac{1}{2}}} \int_{\mathbb{R}^d} \frac{|\mathbf{I}+\mathbf{B}|^{\frac{1}{2}}}{\sqrt{(2\pi)^d}} \exp\big[-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu})'(\mathbf{I}+\mathbf{B})(\mathbf{z}-\boldsymbol{\mu})\big]\, d\mathbf{z} \\ &= \frac{\exp\big[\frac{1}{2}\boldsymbol{\mu}'(\mathbf{I}+\mathbf{B})\boldsymbol{\mu}\big]}{|\mathbf{I}+\mathbf{B}|^{\frac{1}{2}}} \int_{\mathbb{R}^d} \mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}, (\mathbf{I}+\mathbf{B})^{-1})\, d\mathbf{z} \\ &= \frac{\exp\big[\frac{1}{2}\boldsymbol{\mu}'(\mathbf{I}+\mathbf{B})\boldsymbol{\mu}\big]}{|\mathbf{I}+\mathbf{B}|^{\frac{1}{2}}} \\ &= \frac{\exp\big[\frac{1}{2}\mathbf{a}'\boldsymbol{\mu}\big]}{|\mathbf{I}+\mathbf{B}|^{\frac{1}{2}}} \end{aligned} \tag{5.5}$$

Both $\boldsymbol{\mu}$ and the determinant $|\mathbf{I} + \mathbf{B}|$ can be found by Cholesky decomposition of the positive definite matrix $\mathbf{I} + \mathbf{B}$.

Our pooling and expectation operations can now be combined for LR calculation. For f_1, f_2 represented by $(\mathbf{a}_1, \mathbf{B}_1)$ and $(\mathbf{a}_2, \mathbf{B}_2)$, the normalized inner product is given by:

$$\big\langle \overline{f_1}, \overline{f_2} \big\rangle = \frac{\big\langle f_1 f_2 \big\rangle}{\big\langle f_1 \big\rangle \big\langle f_2 \big\rangle} = \frac{E(\mathbf{a}_1 + \mathbf{a}_2, \mathbf{B}_1 + \mathbf{B}_2)}{E(\mathbf{a}_1, \mathbf{B}_1) E(\mathbf{a}_2, \mathbf{B}_2)} \tag{5.6}$$

5.1.4 L2 norm

As we have seen, we can represent all LR calculations via pooling and expectation. For non-negative Gaussians, expectation is also L1 norm. However,

as (3.7) shows, we can alternatively assemble our LR calculations via the L2 norm. For Gaussian meta-embeddings, the L2 norm is readily calculated via the same expectation mechanism derived above. For an embedding f represented by $(\mathbf{a}, \mathbf{B})$, we have:

$$\|f\|^2 = \langle f, f \rangle = \langle f^2 \rangle = \int_{\mathbb{R}^d} f(\mathbf{z})^2 \pi(\mathbf{z})\, d\mathbf{z} = E(2\mathbf{a}, 2\mathbf{B}) \tag{5.7}$$

Notice that

$$\left\|\overline{f}\right\|^2 = \langle \overline{f}, \overline{f} \rangle = \frac{\langle f^2 \rangle}{\langle f \rangle^2} = \frac{E(2\mathbf{a}, 2\mathbf{B})}{E(\mathbf{a}, \mathbf{B})^2} \tag{5.8}$$

If L2 norm is computed using L1 norm, is there any advantage to the L2 norm? If we use the closed-form expressions, maybe not. But as we shall see in section 6.3.2 below, stochastic approximations to the L2 norm may have advantages relative to stochastic L1 norm approximations.

5.1.5 Graphical multivariate Gaussian examples

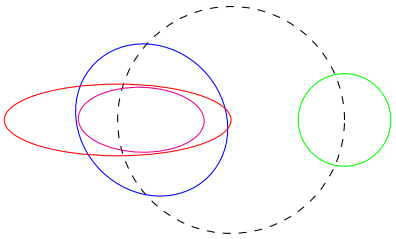
Let us again turn to some simple examples to better understand our multivariate Gaussian meta-embeddings. We choose $\mathbf{z} \in \mathcal{Z} = R^2$, so that we can plot the examples. Recall that in the example of section 4.4 we were able to directly plot points in meta-embedding space. Here, that space is infinite-dimensional, so we cannot plot our meta-embeddings as points. But we *can* plot representations of the Gaussian *distributions* in $\mathcal{Z}$-space.

In our plots below, every meta-embedding is shown as an ellipse, centered at the mean of the Gaussian. The ellipse represents the $\sigma = 1$ contour so that it is elongated in directions where the meta-embedding has less certainty about the location of $\mathbf{z}$, while the ellipse is more compressed in directions of greater certainty.[2]

The figure below shows some hypothetical meta-embeddings and the LRs between them, computed using (5.5) in (4.1), or (4.2). The standard normal prior is shown in dashed black. Blue, red and green represent raw embeddings, while magenta represents blue and red pooled, where pooling is done

[2]The ellipse axes are aligned with the eigenvectors of the covariance, and the radii are the square roots of the eigenvalues. MATLAB code that computes LLRs and outputs Ti*k*Z code for plotting the ellipses in LaTeX is available here: http://github.com/bsxfan/meta-embeddings/tree/master/code/Niko/matlab.

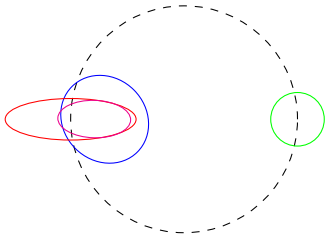
by adding natural parameters ($\mathbf{a}_{\text{mag}} = \mathbf{a}_{\text{blue}} + \mathbf{a}_{\text{red}}$, and $\mathbf{B}_{\text{mag}} = \mathbf{B}_{\text{blue}} + \mathbf{B}_{\text{red}}$):



$\langle \overline{f}_i, \overline{f}_j \rangle$	blue	red	magenta
green	0.37	0.80	0.23
blue		2.34	

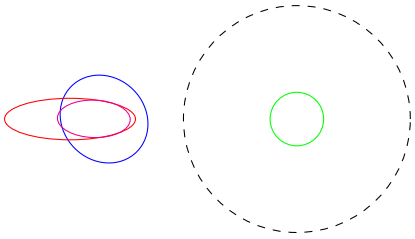
We can imagine that green was extracted from a long, clean recording; blue from a long noisy recording; and red from a short, clean recording. The short red recording gives reasonable certainty along the vertical axis, but crucial information is missing, resulting in greater uncertainty on the horizontal axis. Notice that $\text{LR}(\text{blue}, \text{green}) < \text{LR}(\text{red}, \text{green})$, even though the blue mean is closer to green than the red mean. This is due to the greater red uncertainty in the horizontal direction. Pooling (magenta) dramatically reduces the uncertainty about the location of $\mathbf{z}$ for the speaker represented by red and blue and consequently increases the certainty that this speaker is *not* the same as the green speaker: $\text{LR}(\text{magenta}, \text{green})$ is smaller than all the other LRs. There is enough overlap between red and blue to give some support, $\text{LR}(\text{red}, \text{blue}) > 1$, for the hypothesis that red and blue are of the same speaker.

Next, we repeat the example, keeping everything the same, except that all the embeddings have smaller uncertainty. The effect is that the LRs also become more certain, moving further away from the neutral value of 1:



$\langle \overline{f}_i, \overline{f}_j \rangle$	blue	red	magenta
green	0.008	0.069	7.4e-4
blue		5.264	

Finally, we change the example by shifting the meta-embeddings left, relative to the prior. This affects all the LRs:

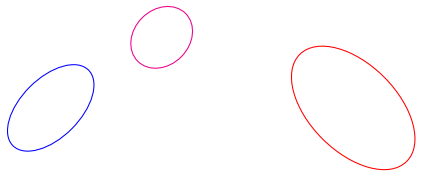


$\langle \overline{f}_i, \overline{f}_j \rangle$	blue	red	magenta
green	0.015	0.162	0.0013
blue		14.260	

The most dramatic effect of the above shift is on LR(red, blue), which has increased threefold due to the increased rarity of the red and blue embeddings, relative to the prior.

Food for thought

Let us examine pooling in more detail. In the above examples, when we pooled the suboptimal information extracted by the (short) red and (noisy) blue recordings, the result (magenta) gave less uncertainty about the value of $\mathbf{z}$ for the speaker common to red and blue. Intuition suggests that things are working properly. But now consider pooling red and blue embeddings which are far apart:



$$\text{LR}(\text{blue}, \text{red}) = 0.0047$$

This result (magenta) is perhaps counter-intuitive:

- Why is the magenta mean so far away from the line connecting the red and blue means?
- If the same speaker is observed in two relatively distant locations, should this not cause the resulting uncertainty to *increase*, rather than the decrease we see here?

Nevertheless, we have followed the theory and the magenta result is correct. One reason why our intuition has a hard time to accept this result is because pooling is conditioned on the assumption that the speakers are the same, yet $\text{LR}(\text{blue}, \text{red}) \ll 1$ gives strong support to the hypothesis that this is not so. If we were to encounter this situation in practice, it would suggest that the same-speaker assumption is wrong, or that the uncertainties represented in the meta-embeddings have been underestimated.

The apparently wayward magenta mean is the location where the blue and red distributions 'overlap most'. It is located at the maximum of the product of the two distributions. It can be interpreted as a soft intersection between the two distributions.

5.2 MoGME: Mixture of GME

A mixture of Gaussians gives a more powerful representation that retains some of the advantages of the GME. For a moderate number of mixture

components, we retain tractability of LR calculations, but pooling is not so straight-forward.

5.2.1 Definition and normalization

Let $\overline{\text{GME}}_{jk}$ denote a *normalized* Gaussian meta-embedding,[3] with parameters indexed by jk. We define a *mixture of Gaussian meta-embeddings* (MoGME) of order K as:

$$f_j(\mathbf{z}) = \sum_{k=1}^{K} w_{jk} \overline{\text{GME}}_{jk}(\mathbf{z}) \tag{5.9}$$

where every $w_{jk} > 0$. As with any meta-embedding, the scaling is unimportant, so we do not insist that $\sum_{k=1}^{K} w_{jk} = 1$. However, the *relative* scaling of the components *is* important, so we do insist on having normalized components. Normalization of the whole MogME is done by weight normalization:

$$\overline{f_j} = \frac{f_j}{\sum_{k=1}^{K} w_{jk}} \tag{5.10}$$

5.2.2 Likelihood ratio

The LR, or normalized inner product, can be computed using the GME inner product (5.6):

$$\begin{aligned} \left\langle \overline{f_i}, \overline{f_j} \right\rangle &= \frac{\left\langle (\sum_{k=1}^{K} w_{ik} \overline{\text{GME}}_{ik})(\sum_{\ell=1}^{K} w_{j\ell} \overline{\text{GME}}_{j\ell}) \right\rangle}{(\sum_{k=1}^{K} w_{ik})(\sum_{\ell=1}^{K} w_{j\ell})} \\ &= \frac{\sum_{k=1}^{K} \sum_{\ell=1}^{K} w_{ik} w_{j\ell} \left\langle \overline{\text{GME}}_{ik}, \overline{\text{GME}}_{j\ell} \right\rangle}{(\sum_{k=1}^{K} w_{ik})(\sum_{\ell=1}^{K} w_{j\ell})} \end{aligned} \tag{5.11}$$

The complexity scales as K^2.

5.2.3 Pooling

Pooling (product) of two MoGMEs of order K gives a MoGME of order K^2, where the resultant weights have an interesting form. To derive the weight

[3] Remember, a normalized meta-embedding is *not* a normalized probability distribution. In fact, it need not even be normalizable. The meta-embedding is normalized when its product with the prior is a normalized distribution.

formula, notice that for any normalized meta-embeddings, $\overline{f}, \overline{g}$, renormalizing their product gives:

$$\overline{\overline{f}\overline{g}} = \frac{\overline{f}\overline{g}}{\langle \overline{f}\overline{g}, \mathbf{1} \rangle} = \frac{\overline{f}\overline{g}}{\langle \overline{f}, \overline{g} \rangle} \tag{5.12}$$

We use this to write the MogME product in the required form with normalized components:

$$\begin{aligned} f_i f_j &= \sum_{k=1}^{K} \sum_{\ell=1}^{K} w_{ik} w_{j\ell} \overline{\mathrm{GME}}_{ik} \overline{\mathrm{GME}}_{j\ell} \\ &= \sum_{k=1}^{K} \sum_{\ell=1}^{K} \Big(w_{ik} w_{j\ell} \langle \overline{\mathrm{GME}}_{ik}, \overline{\mathrm{GME}}_{j\ell} \rangle \Big) \overline{\overline{\mathrm{GME}}_{ik} \overline{\mathrm{GME}}_{j\ell}} \end{aligned} \tag{5.13}$$

where the parenthesis shows the new weight for component $k\ell$. Pairs of matching components get large weights, while mismatched pairs get smaller weights.

To avoid the polynomial explosion of complexity associated with pooling, the possibility exists to form approximate products that retain only K terms. The weights could be used to select the best K. More sophisticated methods could numerically optimize the normalized inner product between full and approximate representations, but this would have K^3 complexity per iteration.

5.3 Exponential family Gaussian mixture

The multivariate Gaussian of the previous section is of course an exponential family distribution. We can derive very similar recipes—also with logarithmic representations—from other exponential families. We develop one such example here.

Although a Gaussian mixture is not an exponential family, the *joint* distribution for the continuous and discrete (state) variable *is* exponential family. We do this with a D-component mixture of d-dimensional Gaussians, by choosing our hidden speaker identity variable as: $\mathbf{z} = (\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, \ldots, x_D)$ is a one-hot vector of size D, while $\mathbf{y} \in \mathbb{R}^d$. Our meta-embedding for recording j is:

$$f_j(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{D} \Big(w_{ij} \exp\big[\mathbf{a}'_{ij} \mathbf{y} - \frac{1}{2} \mathbf{y}' \mathbf{B}_{ij} \mathbf{y} \big] \Big)^{x_i} \tag{5.14}$$

where $x_i \in \{0, 1\}$, $w_{ij} > 0$, $\mathbf{a}_{ij} \in \mathbb{R}^d$ and the $\mathbf{B}_{ij}$ are d-by-d positive semi-definite. To see that this is exponential family, take the logarithm and re-arrange:

$$\log f_j(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{D} x_i \log w_{ij} + (x_i \mathbf{y}')\mathbf{a}_{ij} - \frac{1}{2} \operatorname{tr}\big[(x_i \mathbf{y}\mathbf{y}')\mathbf{B}_{ij}\big] \tag{5.15}$$

where we see the sufficient statistics are: $\{x_i, x_i\mathbf{y}, x_i\mathbf{y}\mathbf{y}'\}_{i=1}^{D}$ and the natural parameters are $\{\log w_{ij}, \mathbf{a}_{ij}, \mathbf{B}_{ij}\}_{i=1}^{D}$. To form more compact representations, we can let the natural parameters be linear functions of smaller vectors (of which some components have to be constrained to be non-negative). As above, elementwise multiplication (pooling) is accomplished by simple vector addition of these representations.

5.3.1 Prior

We choose a parameterless prior, of the same form as the meta-embeddings:

$$\pi(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{D} \Big(\frac{\mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{I})}{D} \Big)^{x_i} \tag{5.16}$$

and

$$\begin{aligned} \log \pi(\mathbf{x}, \mathbf{y}) &= \sum_{i=1}^{D} x_i \big(-\frac{1}{2}\mathbf{y}'\mathbf{y} - \frac{d}{2}\log(2\pi) - \log(D) \big) \\ &= -\frac{d}{2}\log(2\pi) - \log(D) + \sum_{i=1}^{D} -\frac{1}{2} \operatorname{tr}\big[(x_i \mathbf{y}\mathbf{y}')\mathbf{I}\big] \end{aligned} \tag{5.17}$$

As in the multivariate Gaussian case, the practical function of the prior is to add $\mathbf{I}$ to the possibly semi-definite matrices, $\mathbf{B}_{ij}$, of the meta-embeddings, to make them properly positive definite.

5.3.2 Expectation

Dropping the subscript j, let $f(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{D} \Big(w_i \exp\big[\mathbf{a}_i'\mathbf{y} - \frac{1}{2}\mathbf{y}'\mathbf{B}_i\mathbf{y}\big] \Big)^{x_i}$, then:[4]

$$\begin{aligned} E\big(\{w_i, \mathbf{a}_i, \mathbf{B}_i\}_{i=1}^{D}\big) &= \big\langle f \big\rangle \\ &= \sum_{i=1}^{D} \frac{w_i}{D} \int_{\mathbb{R}^d} f(\mathbf{x} = i, \mathbf{y}) \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{I})\, d\mathbf{y} \\ &= \sum_{i=1}^{D} \frac{w_i}{D} \int_{\mathbb{R}^d} \exp\big[\mathbf{a}_i'\mathbf{y} - \frac{1}{2}\mathbf{y}'\mathbf{B}_i\mathbf{y}\big] \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{I})\, d\mathbf{y} \\ &= \sum_{i=1}^{D} \frac{w_i}{D} \frac{\exp\big[\frac{1}{2}\mathbf{a}_i'\boldsymbol{\mu}_i\big]}{|\mathbf{I} + \mathbf{B}_i|^{\frac{1}{2}}} \end{aligned} \tag{5.18}$$

where $\boldsymbol{\mu}_i = (\mathbf{I} + \mathbf{B}_i)^{-1}\mathbf{a}_i$.

This computation is much the same as for the multivariate Gaussian case, except that we have to do D such calculations every time. Obviously, if we want to use a large value for D, then the individual calculations need to be cheap. Since Cholesky decomposition of full matrices $\mathbf{I} + \mathbf{B}_i$ requires $\mathcal{O}(d^3)$ computation, we need to simplify these calculations. There are various way to do that. We can let the $\mathbf{B}_{ij}$ differ from each other by scaling, by low-rank modifications, or by forming them via Kronecker products, etc.

5.4 Mixtures

Let us compare the previous solution with one where the state is not considered part of the speaker identity variable. The meta-embedding for recording r_j is:

$$f_j(\mathbf{z}) = \sum_{i=1}^{D} w_{ij} \tag{5.19}$$

[4] We abuse notation by writing $\mathbf{x} = i$ to indicate that component i is the hot element (value 1) in the otherwise zero one-hot vector, $\mathbf{x}$.

5.5 Free form, inspired by exponential family distribution

5.6 Discrete Factorial

5.7 Mixture with fixed components

5.8 Mixture with shifted components

5.9 Kernel approximation

Let us think in terms of inner products, rather than expectations. If we want fast LR computation, we need fast inner product computation.

5.10 Mean embedding

Chapter 6

Approximate computation

Meta-embeddings are more powerful than existing methods and therefore potentially more accurate, but the required calculation of inner products or expectations may be too slow to be useful in practice, unavailable in closed form, and maybe even completely intractable. Even the multivariate Gaussian meta-embeddings, which have tractable closed-form calculations are a lot slower than traditional Gaussian PLDA (GPLDA) scoring. In this chapter we examine some proposals for fast approximations. For reference below, we briefly derive the GPLDA scoring function.

6.1 The GPLDA score

When recordings are represented by i-vectors, which are in turn modelled by Gaussian PLDA, then a Gaussian meta-embedding (GME), with *constant precision*, can be extracted in closed form from every i-vector. In this case, no neural net needs to be trained to extract the meta-embeddings. Each meta-embedding is extracted with a simple function of the i-vector and the GPLDA parameters. (This formula is given by (7.7) below, by letting $\nu \to \infty$.) The GPLDA parameters can be generatively, or discriminatively trained.

Following the notation of section 5.1, the GPLDA GMEs have $D = 0$, giving constant precision, $\mathbf{B}_j = \mathbf{E}_0$. This leads to a fast, quadratic scoring recipe. Let the embeddings f_i, f_j be represented by $\mathbf{a}_i$ and $\mathbf{a}_j$, then by (5.6) the log-LR between them is:

$$\begin{aligned}\log\langle \overline{f_i}, \overline{f_j} \rangle &= \log E(\mathbf{a}_i + \mathbf{a}_j, 2\mathbf{E}_0) - \log E(\mathbf{a}_i, \mathbf{E}_0) - \log E(\mathbf{a}_j, \mathbf{E}_0) \\ &= (\mathbf{a}_i + \mathbf{a}_j)'\mathbf{K}(\mathbf{a}_i + \mathbf{a}_j) - \mathbf{a}_i'\mathbf{L}\mathbf{a}_i - \mathbf{a}_j'\mathbf{L}\mathbf{a}_j + C\end{aligned} \tag{6.1}$$

where C is a constant representing the log-determinant terms and

$$\mathbf{K} = (\mathbf{I} + 2\mathbf{E}_0)^{-1} \qquad \text{and} \qquad \mathbf{L} = (\mathbf{I} + \mathbf{E}_0)^{-1} \tag{6.2}$$

6.2 Taylor series approximations

In this section, we assume that we have available a slow, but closed-form scoring recipe (e.g. GME) to compute the log-LR, $\log\langle \overline{f}, \overline{g} \rangle$. Second-order Taylor series approximations of $\log\langle \overline{f}, \overline{g} \rangle$ will give us scoring recipes with the same functional form—and therefore the same speed—as the quadratic GPLDA score above.

Let f, g be represented respectively by $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$ and let the scoring function be denoted:

$$F(\mathbf{x}, \mathbf{y}) = \log\langle \overline{f}, \overline{g} \rangle \tag{6.3}$$

We assume that F can be evaluated and differentiated at least twice. There are at least two ways to form second-order Taylor series approximations. We consider symmetric and asymmetric variants.

6.2.1 Symmetric Taylor series approximation

Define $\mathbf{z}$ as the stacked vector containing both sides: $\mathbf{z} = \begin{bmatrix} \mathbf{x}' & \mathbf{y}' \end{bmatrix}'$. We do the second-order expansion about some convenient, fixed value $\mathbf{z}_0 = \begin{bmatrix} \mathbf{x}_0' & \mathbf{y}_0' \end{bmatrix}'$. The approximation is:

$$\begin{aligned} F(\mathbf{x}, \mathbf{y}) \approx\ & F(\mathbf{x}_0, \mathbf{y}_0) + (\mathbf{z} - \mathbf{z}_0)' \nabla_{\mathbf{z}}(\mathbf{x}_0, \mathbf{y}_0) \\ & + \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)' \mathbf{H}_{\mathbf{z}}(\mathbf{x}_0, \mathbf{y}_0)(\mathbf{z} - \mathbf{z}_0) \end{aligned} \tag{6.4}$$

where $\nabla_{\mathbf{z}}$ and $\mathbf{H}_{\mathbf{z}}$ are gradient and Hessian of F, obtained by differentiating w.r.t. $\mathbf{z}$. Since $\mathbf{z}_0$ is fixed, all of $F(\mathbf{x}_0, \mathbf{y}_0)$, $\nabla_{\mathbf{z}}(\mathbf{x}_0, \mathbf{y}_0)$ and $\mathbf{H}_{\mathbf{z}}(\mathbf{x}_0, \mathbf{y}_0)$ can be precomputed before runtime. The storage requirement is $1 + k + k^2$, or a bit smaller if the symmetry of the Hessian is used.

It is easy to verify that when this approximation is applied to the PLDA score (6.1), at for example $\mathbf{z}_0 = \mathbf{0}$, we recover the GPLDA score exactly.

Implementation

In an environment where both complex arithmetic and first-order derivatives of F are available, *complex-step automatic differentiation* can be used (with some caveats) to compute the Hessian to high accuracy, without having to resort to laborious hand-derived second-order differentiation.

6.2.2 Asymmetric Taylor series approximation

A potentially more accurate, but somewhat slower approximation can be obtained by evaluating $F(\mathbf{x}, \mathbf{y})$ at the exact, required value of say $\mathbf{x}$ and doing the expansion only w.r.t. $\mathbf{y}$. The approximation is:

$$\begin{aligned} F(\mathbf{x}, \mathbf{y}) \approx F(\mathbf{x}, \mathbf{y}_0) + (\mathbf{y} - \mathbf{y}_0)' \nabla_{\mathbf{y}}(\mathbf{x}, \mathbf{y}_0) \\ + \frac{1}{2}(\mathbf{y} - \mathbf{y}_0)' \mathbf{H}_{\mathbf{y}}(\mathbf{x}, \mathbf{y}_0)(\mathbf{y} - \mathbf{y}_0) \end{aligned} \tag{6.5}$$

This recipe is slower, because for every $\mathbf{x}$ we need a new computation of $F(\mathbf{x}, \mathbf{y}_0)$, $\nabla_{\mathbf{y}}(\mathbf{x}, \mathbf{y}_0)$ and $\mathbf{H}_{\mathbf{y}}(\mathbf{x}, \mathbf{y}_0)$. Of course, this recipe can be symmetrized by doing it also from the other side and averaging.

6.3 Stochastic approximations

Since our meta-embeddings are essentially probability distributions, we can sample from them in many cases. This is explored below.

6.3.1 Stochastic expectation for GMEs

For the GMEs of section 5.1, pooling is fast (addition), but expectation is slower (Cholesky decomposition). Unfortunately, when precisions vary, this representation requires a new Cholesky decomposition for *every* LR calculation.

In applications where many verification trials are processed on limited hardware, this will have a significant impact on speed. Even if we have enough CPU power available at runtime for some applications, discriminative training of meta-embedding extractors remains a problem because discriminative training criteria typically require evaluating very many trials. Let us therefore consider some plans for speeding up scoring of trials by making stochastic approximations.

We envisage that such stochastic approximation could work out to be especially cheap at training time, in the same way that stochastic evaluation of the ELBO in *variational autoencoder* (VAE) training typically requires but a single stochastic sample per training example [30]. Given sufficiently many examples, the stochastic approximation errors tend to cancel, provided the errors are independent.

We will not be too concerned about expectations of a single factor, of the form $\langle f \rangle$. These expectations are required to normalize trial sides (typically meta-embeddings extracted from single recordings) and there are far fewer

trial sides than there are trials. The calculations for which we need speed, are expectations of the form $\langle \overline{f}_i \, \overline{f}_j \rangle$. We consider two proposals below.

Prior sampling

An obvious choice, perhaps naive, would be to form the expectation by sampling from the prior, π. The expectation could be approximated via N such samples as:

$$\left\langle \overline{f}_i(\mathbf{z})\overline{f}_j(\mathbf{z}) \right\rangle_{\mathbf{z}\sim\pi} \approx \frac{1}{N} \sum_{\tilde{\mathbf{z}}\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \overline{f}_i(\tilde{\mathbf{z}})\overline{f}_j(\tilde{\mathbf{z}}) \tag{6.6}$$

We fear this might be inaccurate, because in high-dimensional space, samples from π will be very unlikely to hit the peak[1] of $\overline{f}_i \, \overline{f}_j$ and therefore an affordable number of samples might not be able to sufficiently explore the volume under the peak. Experiments might be needed to check whether this is indeed a problem. Let us nevertheless explore in more detail the calculation required for each sample:

$$\begin{aligned} \overline{f}_i(\tilde{\mathbf{z}})\overline{f}_j(\tilde{\mathbf{z}}) &= c_i c_j \exp\left[\mathbf{a}_i'\tilde{\mathbf{z}} + \mathbf{a}_j'\tilde{\mathbf{z}} - \frac{1}{2}\tilde{\mathbf{z}}'\mathbf{B}_i\tilde{\mathbf{z}} - \frac{1}{2}\tilde{\mathbf{z}}'\mathbf{B}_j\tilde{\mathbf{z}}\right] \\ &= c_i c_j \exp\left[\mathbf{a}_i'\tilde{\mathbf{z}} + \mathbf{a}_j'\tilde{\mathbf{z}} - \frac{1}{2}\mathbf{b}_i'\tilde{\boldsymbol{\gamma}} - \frac{1}{2}\mathbf{b}_j'\tilde{\boldsymbol{\gamma}}\right] \end{aligned} \tag{6.7}$$

where using (5.2), we have defined $\tilde{\boldsymbol{\gamma}} = [\tilde{\gamma}_1, \ldots, \tilde{\gamma}_D] \in \mathbb{R}^D$, with $\tilde{\gamma}_\ell = \tilde{\mathbf{z}}'\mathbf{E}_\ell\tilde{\mathbf{z}}$ and where c_i, c_j reflect the normalizations. The meta-embedding representations, $\mathbf{a}_i, \mathbf{b}_i$ and $\mathbf{a}_j, \mathbf{b}_j$, and the trainable constants, $\{\mathbf{E}_\ell\}_{\ell=1}^D$, are as defined in the beginning of section 5.1.

Notice that $\tilde{\boldsymbol{\gamma}}$ is independent of the meta-embeddings and therefore independent of the data and can be precomputed. If the $\mathbf{E}_\ell$ are constrained to be either diagonal or rank-one, then the calculations $\tilde{\mathbf{z}}'\mathbf{E}_\ell\mathbf{z}$ will be cheap. Keep in mind that we need m different samples, $\tilde{\mathbf{z}}$, and therefore also N different versions of $\tilde{\boldsymbol{\gamma}}$. In stochastic minibatch training, if we update the $\{\mathbf{E}_\ell\}$ after every minibatch, we will have to update all of the $\tilde{\boldsymbol{\gamma}}$ also, in which case it might make sense to also resample N new values for $\tilde{\mathbf{z}}$. This type of strategy, which is well-known also in VAE [30], is termed *doubly stochastic* by Michalis Titsias [31].

An advantage of (6.7) is that it generalizes to expectations of more (or fewer) than two factors—the argument in the exponent will have independent terms for each of the factors in the expectation.

[1] The product is still Gaussian and therefore has single peak.

Experiments would have to be conducted to verify if an affordably small m could give reasonable accuracy. Keep in mind that in such experiments, we can compare accuracy against the exact calculation (5.5).

Posterior sampling

If meta-embedding f_i represents recording r_i, then

$$\pi(\mathbf{z})\overline{f_i}(\mathbf{z}) = P(\mathbf{z} \mid r_i, \pi)$$

is a properly normalized Gaussian, from which we can sample.[2] We can now rewrite the expectation as:

$$\left\langle \overline{f_i}(\mathbf{z})\overline{f_j}(\mathbf{z}) \right\rangle_{\mathbf{z}\sim\pi} = \left\langle \overline{f_j}(\tilde{\mathbf{z}}) \right\rangle_{\tilde{\mathbf{z}}\sim\pi\overline{f_i}} \approx \frac{1}{N} \sum_{\tilde{\mathbf{z}}\sim\pi\overline{f_i}} c_j \exp\left[\mathbf{a}_j'\tilde{\mathbf{z}} - \frac{1}{2}\mathbf{b}_j'\tilde{\boldsymbol{\gamma}}\right] \tag{6.8}$$

This might be more accurate than prior sampling, because when f_i and f_j represent the same speaker, their peaks should overlap. If they are of different speakers, their peaks might be far apart, but then we do want a value close to zero for the expectation. With this variant, we may be able to use smaller values for N.

This method is asymmetric. Which version is best—should we sample from f_i, or from f_j? If f_i is much sharper than f_j, so that f_j is almost constant compared to f_j, then one sample would suffice, while if we sampled the other way round, we would need many samples to properly explore the volume of f_i. Pooling generally sharpens peaks, so it would be more accurate to sample from the side that consists of the most pooled factors. Unfortunately, fast pooling and sampling are conflicting requirements—after adding natural parameters to compute the pooling, we need a Cholesky factorization before we can sample.

Whether we pool or not, a disadvantage of posterior sampling vs prior sampling is that we have reintroduced the need for Cholesky factorization. Fortunately, we need this only for one side of the trial and we are still avoiding per-trial Cholesky decompositions. Also note that for the sampled side of the trial we do not need to separately compute the normalization constant, because the samples already come from the normalized distribution.

Another option would be to change the representation. We could let:

$$\mathbf{B}_i = \Big(\sum_{\ell=1}^{D} b_{\ell i}\mathbf{T}_\ell\Big)\Big(\sum_{\ell=1}^{D} b_{\ell i}\mathbf{T}_\ell\Big)' \tag{6.9}$$

[2]The same is not always true of f_i or $\overline{f_i}$, which might have a singular precision, in which case sampling could not be done.

where the $\mathbf{T}_\ell$ are triangular and the $b_{\ell i}$ are no longer constrained to be non-negative. This gives a free Cholesky decomposition, and fast, approximate evaluation of LRs of the form $\langle \overline{f}_i, \overline{f}_j \rangle$, but it complicates pooling for more complex LR calculations.

6.3.2 Stochastic L2 norm

The above stochastic expectation recipes are fragile because the distributions from which we sample can be mismatched to the arguments of the expectations. If we instead do stochastic L2 norm, we can make use of the symmetry to combat this problem. The analysis in this section is applicable more generally, not just to multivariate Gaussian meta-embeddings.

Using (3.7), we want to do the following calculation:

$$\langle \overline{f}, \overline{g} \rangle = \frac{1}{2}\left(\left\|\overline{f} + \overline{g}\right\|^2 - \left\|\overline{f}\right\|^2 - \|\overline{g}\|^2\right) \tag{6.10}$$

Again, we assume that we can afford to compute each $\left\|\overline{f}\right\|^2$ and $\|\overline{g}\|^2$ in closed form, using (5.8). However, we need a faster way to compute $\left\|\overline{f} + \overline{g}\right\|^2$. Keep in mind that $\pi(\mathbf{z})\overline{f}(\mathbf{z})$ and $\pi(\mathbf{z})\overline{g}(\mathbf{z})$ are normalized probability densities from which we can sample. We can also sample from the *mixture*:

$$\mathcal{M}(\mathbf{z}) = \frac{1}{2}\pi(\mathbf{z})\left(\overline{f}(\mathbf{z}) + \overline{g}(\mathbf{z})\right) \tag{6.11}$$

Expanding the term of interest, we find:

$$\begin{aligned}
&\frac{1}{2}\left\|\overline{f} + \overline{g}\right\|^2 \\
&= \int_{\mathbb{R}^d} \frac{1}{2}\pi(\mathbf{z})\left(\overline{f}(\mathbf{z}) + \overline{g}(\mathbf{z})\right)^2 d\mathbf{z} \\
&= \int_{\mathbb{R}^d} \mathcal{M}(\mathbf{z})\left(\overline{f}(\mathbf{z}) + \overline{g}(\mathbf{z})\right) d\mathbf{z} \\
&= \frac{1}{2}\int_{\mathbb{R}^d} \pi(\mathbf{z})\overline{f}(\mathbf{z})\left(\overline{f}(\mathbf{z}) + \overline{g}(\mathbf{z})\right) d\mathbf{z} + \frac{1}{2}\int_{\mathbb{R}^d} \pi(\mathbf{z})\overline{g}(\mathbf{z})\left(\overline{f}(\mathbf{z}) + \overline{g}(\mathbf{z})\right) d\mathbf{z} \\
&\approx \frac{1}{N}\sum_{\tilde{\mathbf{z}}\sim\pi\overline{f}} \frac{\overline{f}(\tilde{\mathbf{z}}) + \overline{g}(\tilde{\mathbf{z}})}{2} + \frac{1}{N}\sum_{\tilde{\mathbf{z}}\sim\pi\overline{g}} \frac{\overline{f}(\tilde{\mathbf{z}}) + \overline{g}(\tilde{\mathbf{z}})}{2}
\end{aligned} \tag{6.12}$$

where N samples are drawn from each distribution. If f and g are both sharp relative to the prior, then the sampling distribution and argument of

the expectation are very well matched. Even if f or g are not so sharp, then the peaks of $\mathcal{M}(\mathbf{z})$ will be sharper, which is the good way round.

In a variant of this recipe, we could also compute $\left\|\overline{f}\right\|^2$ and $\|\overline{g}\|^2$ stochastically. If we use the same samples everywhere, cancellation of terms gives:

$$\begin{aligned}\langle \overline{f}, \overline{g} \rangle &= \frac{1}{2}\left\|\overline{f}+\overline{g}\right\|^2 - \frac{1}{2}\left\|\overline{f}\right\|^2 - \frac{1}{2}\|\overline{g}\|^2 \\ &\approx \frac{1}{N}\sum_{\tilde{\mathbf{z}}\sim\pi\overline{f}} \frac{\overline{g}(\tilde{\mathbf{z}})}{2} + \frac{1}{N}\sum_{\tilde{\mathbf{z}}\sim\pi\overline{g}} \frac{\overline{f}(\tilde{\mathbf{z}})}{2}\end{aligned} \tag{6.13}$$

which is just a symmetrized application of (6.8). If the matched sampling recipe (6.12) is beneficial, then the above symmetrized cross-sampling should presumably enjoy the same benefits.

6.4 Laplace approximation for pooling

Here we show how the *Laplace approximation* [32, 33] can be applied to continuous unimodal meta-embeddings, where closed-form pooling is unavailable, but where we can afford an iterative calculation to find an approximate pooled meta-embedding. An example would be the t-distribution meta-embeddings which will be discussed in section 7.1 below.

For meta-embeddings, $f_1, \ldots, f_n$, that we want to pool, define the function:

$$F(\mathbf{z}) = -\sum_{i=1}^{n} \log f_i(\mathbf{z}) \tag{6.14}$$

Use numerical optimization to find $\hat{\mathbf{z}} = \operatorname{argmin} F(\mathbf{z})$. Find the Hessian, $\mathbf{H}$, at $\hat{\mathbf{z}}$. The gradient should be zero, so that the second-order Taylor-series approximation about $\hat{\mathbf{z}}$ is:

$$F(\mathbf{z}) \approx F(\hat{\mathbf{z}}) + \frac{1}{2}(\mathbf{z}-\hat{\mathbf{z}})'\mathbf{H}(\mathbf{z}-\hat{\mathbf{z}}) \tag{6.15}$$

Exponentiating this gives a Gaussian, which is the Laplace approximation to the pooled meta-embedding:

$$e^{-F(\mathbf{z})} \approx \exp[-\frac{1}{2}(\mathbf{z}-\hat{\mathbf{z}})'\mathbf{H}(\mathbf{z}-\hat{\mathbf{z}}) + \text{const}] \tag{6.16}$$

6.5 Multiclass classification

Multiclass classification can be scored with (4.3). The likelihood that a test recording, r_j, represented by meta-embedding f_j belongs to class i, for which we have as training examples a set of meta-embeddings, $\mathcal{F}_i$, is:

$$L_{ij} = \langle \overline{g_j}, \overline{f_i} \rangle, \qquad \text{where} \qquad f_i = \prod_{f \in \mathcal{F}_i} f \tag{6.17}$$

Here we are interested in the case where for every class i, we have available a large set of recordings, so that the uncertainty in f_i becomes very small. For the continuous case, we assume that $\overline{f_i}(\mathbf{z})\pi(\mathbf{z}) \approx \delta(\mathbf{z} - \hat{\mathbf{z}}_i)$, where δ denotes the Dirac delta impulse. In this case we have:

$$L_{ij} \approx \int_{\mathcal{Z}} \overline{f_j}(\mathbf{z})\delta(\mathbf{z} - \hat{\mathbf{z}}_i)\, d\mathbf{z} = \overline{f_j}(\hat{\mathbf{z}}_i) \tag{6.18}$$

For discrete $\mathcal{Z}$ we get the same result, if f_i concentrates at a single element of $\mathcal{Z}$.

6.5.1 Open-set vs closed-set

For N classes, each of which has a large training set, we can use the above L_{ij} as-is for closed-set classification scores. For open-set classification, all we need is another class with no training data. For that class, say $i = N+1$, we would have $f_{N+1} = \mathbf{1}$ and when scored against test g_j, we get $L_{(N+1)j} = \langle \overline{g_j}, \mathbf{1} \rangle = 1$. (Note that here we do not use the approximation, because in this case, f_{N+1} does not concentrate—in fact, it gives equal likelihood to all elements.)

For the open-set case, since we are comparing all the other likelihoods to $L_{(N+1)j} = 1$, the normalization is important, so we need: $L_{ij} \approx \overline{f_j}(\hat{\mathbf{z}}_i)$. On the other hand, in the closed-set case, all likelihoods can be scaled by an arbitrary factor dependent on j, but independent of i. In the closed-set case, we can use $L_{ij} \approx \overline{f_j}(\hat{\mathbf{z}}_i) \propto f_j(\hat{\mathbf{z}}_i)$ if that turns out to be more convenient.

Caveat

A caveat is in order. For the open-set score to work properly in practice, we need a good prior, $\pi(\mathbf{z})$. In a PLDA speaker recognizer for D-dimensional i-vectors, and $\mathcal{Z} = \mathbb{R}^d$, where $d < D$, the prior effectively spans a d-dimensional subspace which best represents all of the speakers seen in training, where the number of training speakers is always greater than d. In spoken language recognition, there are typically very few training languages[34], not enough to span a subspace that would give a good prior for new, unseen languages.

Chapter 7

Meta-embeddings derived from generative models

7.1 PLDA with heavy-tailed noise

In this section, we analyze a variant of heavy-tailed PLDA [35] that is simple enough to allow closed-form derivation of the meta-embeddings, yet complex enough to demonstrate data-dependent uncertainty. Although these meta-embeddings have a closed form, the LRs computed from them do not. However, we show how to construct approximations that could serve as practical alternatives to the popular combination [28] of length-normalized i-vectors with Gaussian PLDA.

Our hidden speaker identity variables live in $\mathcal{Z} = \mathbb{R}^d$. The observed data (representations of the recordings) are i-vector-like and live in $\mathbb{R}^D$. In general, we could allow any $D, d \geq 1$, but we get desirable properties for $D - d \gg 1$. The model described here is somewhat simpler than Kenny's heavy-tailed PLDA of [35], which has hidden speaker and channel factors, both with heavy-tailed priors.

7.1.1 Prior

The identity variable for speaker i is denoted $\mathbf{z}_i \in \mathbb{R}^d$ and they are drawn independently from $N(\mathbf{0}, \mathbf{I})$.

7.1.2 Noise model and likelihood function

The likelihood, $P(\mathbf{r}_j \mid \mathbf{z}_i)$ is parametrized by a rectangular, D-by-d *factor loading matrix*, $\mathbf{F}$, and the *degrees of freedom*, an integer, $\nu \geq 1$. For speaker

i, multiple observations, $\mathbf{r}_j \in \mathbb{R}^D$, are produced by projection into the D-dimensional space and the addition of independently drawn noise, $\boldsymbol{\eta}_j$:

$$\mathbf{r}_j = \mathbf{F}\mathbf{z}_i + \boldsymbol{\eta}_j \tag{7.1}$$

The noise is heavy-tailed in the sense that it is Gaussian with variable precision, λ_j, sampled from the chi-squared distribution, χ^2_ν, parametrized by ν, the *degrees of freedom*. The noise is generated as:

$$\boldsymbol{\eta}_j \sim \mathcal{N}\big(\mathbf{0}, (\frac{\lambda_j}{\nu}\mathbf{W})^{-1}\big) \qquad \text{and} \qquad \lambda_j \sim \chi^2_\nu \tag{7.2}$$

where $\mathbf{W}$ is D-by-D positive definite. The expected value of the precision modulation factor is $\left\langle \frac{\lambda_j}{\nu} \right\rangle = 1$. Marginalizing out the hidden λ_j gives the likelihood, which is a *t-distribution*:

$$\begin{aligned} P(\mathbf{r} \mid \mathbf{z}_i) &= \mathcal{T}(\mathbf{r} \mid \mathbf{F}\mathbf{z}_i, \mathbf{W}^{-1}, \nu) \\ &\propto \Big[1 + \frac{(\mathbf{r} - \mathbf{F}\mathbf{z}_i)'\mathbf{W}(\mathbf{r} - \mathbf{F}\mathbf{z}_i)}{\nu}\Big]^{-\frac{\nu+D}{2}} \end{aligned} \tag{7.3}$$

where we have ignored the somewhat complex normalization constant, which is irrelevant to our present purpose of inference for $\mathbf{z}$. We can use (7.3) as-is for our meta-embedding:

$$f_j(\mathbf{z}) \propto \Big[1 + \frac{(\mathbf{r}_j - \mathbf{F}\mathbf{z})'\mathbf{W}(\mathbf{r}_j - \mathbf{F}\mathbf{z})}{\nu}\Big]^{-\frac{\nu+D}{2}} \tag{7.4}$$

But if $D \geq d$ and $\mathbf{F}'\mathbf{W}\mathbf{F}$ is invertible, we can conveniently rearrange this into a t-distribution for $\mathbf{z}$.

7.1.3 TME: T-distribution meta-embedding

To see that $f_j(\mathbf{z})$ is a t-distrition, we define (dropping subscripts for now):

$$\mathbf{B} = \mathbf{F}'\mathbf{W}\mathbf{F}, \quad \boldsymbol{\mu} = \mathbf{B}^{-1}\mathbf{F}'\mathbf{W}\mathbf{r} \quad \text{and} \quad \mathbf{G} = \mathbf{W} - \mathbf{W}\mathbf{F}\mathbf{B}^{-1}\mathbf{F}'\mathbf{W} \tag{7.5}$$

and we rearrange the quadratic form as:

$$\begin{aligned} Q_{\mathbf{r}|\mathbf{z}} &= (\mathbf{r} - \mathbf{F}\mathbf{z})'\mathbf{W}(\mathbf{r} - \mathbf{F}\mathbf{z}) \\ &= \mathbf{r}'\mathbf{W}\mathbf{r} - 2\mathbf{z}'\mathbf{F}'\mathbf{W}\mathbf{r} + \mathbf{z}'\mathbf{F}'\mathbf{W}\mathbf{F}\mathbf{z} \\ &= \mathbf{r}'\mathbf{W}\mathbf{r} - 2\mathbf{z}'\mathbf{B}\boldsymbol{\mu} + \mathbf{z}'\mathbf{B}\mathbf{z} \\ &= (\mathbf{r}'\mathbf{W}\mathbf{r} - \boldsymbol{\mu}'\mathbf{B}\boldsymbol{\mu}) + (\mathbf{z}'\mathbf{B}\mathbf{z} - 2\mathbf{z}\mathbf{B}\boldsymbol{\mu} + \boldsymbol{\mu}'\mathbf{B}\boldsymbol{\mu}) \\ &= \mathbf{r}'\mathbf{G}\mathbf{r} + (\mathbf{z} - \boldsymbol{\mu})'\mathbf{B}(\mathbf{z} - \boldsymbol{\mu}) \\ &= E + Q_{\mathbf{z}|\mathbf{r}} \end{aligned} \tag{7.6}$$

Reintroducing the subscript j, and defining $\nu' = \nu + D - d$, let's reassemble our t-distribution likelihood:

$$\begin{aligned} f_j(\mathbf{z}) &\propto \Big[1 + \frac{Q_{\mathbf{r}|\mathbf{z}}}{\nu}\Big]^{-\frac{\nu+D}{2}} \\ &\propto \Big[\frac{\nu}{\nu'} + \frac{Q_{\mathbf{r}|\mathbf{z}}}{\nu'}\Big]^{-\frac{\nu'+d}{2}} \\ &= \Big[\frac{\nu}{\nu'} + \frac{E_j + Q_{\mathbf{z}|\mathbf{r}}}{\nu'}\Big]^{-\frac{\nu'+d}{2}} \\ &= \Big[\frac{\nu + E_j}{\nu'} + \frac{Q_{\mathbf{z}|\mathbf{r}}}{\nu'}\Big]^{-\frac{\nu'+d}{2}} \\ &\propto \Big[1 + \frac{(\frac{\nu'}{\nu+E_j})Q_{\mathbf{z}|\mathbf{r}}}{\nu'}\Big]^{-\frac{\nu'+d}{2}} \\ &= \Big[1 + \frac{(\frac{\nu'}{\nu+E_j})(\mathbf{z}-\boldsymbol{\mu}_j)'\mathbf{B}(\mathbf{z}-\boldsymbol{\mu}_j)}{\nu'}\Big]^{-\frac{\nu'+d}{2}} \\ &\propto \mathcal{T}(\mathbf{z} \mid \boldsymbol{\mu}_j, \beta_j\mathbf{B}, \nu') \end{aligned} \tag{7.7}$$

where

$$\boldsymbol{\mu}_j = \mathbf{B}^{-1}\mathbf{F}'\mathbf{W}\mathbf{r}_j\,, \qquad \beta_j = \frac{\nu'}{\nu + E_j} \qquad \text{and} \qquad E_j = \mathbf{r}_j'\mathbf{G}\mathbf{r}_j \tag{7.8}$$

We have shown that the meta-embedding, $f_j(\mathbf{z})$, is proportional to a t-distribution, with *increased* degrees of freedom $\nu' = \nu + (D - d)$, location parameter $\boldsymbol{\mu}_j$ and precision[1] $\beta_j\mathbf{B}$. The mean exists only for $\nu' > 1$ and is then $\boldsymbol{\mu}_j$. The mode is $\boldsymbol{\mu}_j$ and is therefore also the maximum-likelihood estimate for $\mathbf{z}$. We shall refer to the *t-distribution meta-embedding* as TME.

In a typical i-vector PLDA speaker recognizer, we have $400 \leq D \leq 600$, while $100 \leq d \leq 200$. Since $\nu > 0$, this means $\nu' > 200$, which for all practical purposes means that $f_j(\mathbf{z})$ is Gaussian:

$$f_j\mathbf{z} \propto \mathcal{T}(\mathbf{z} \mid \boldsymbol{\mu}_j, \beta_j\mathbf{B}, \nu') \approx \mathcal{N}\big(\mathbf{z} \mid \boldsymbol{\mu}_j, (\beta_j\mathbf{B})^{-1}\big) \tag{7.9}$$

Nevertheless, the almost-Gaussian TME is *substantially* different from the GME given by Gaussian PLDA. In Gaussian PLDA (with $\nu \to \infty$), the meta-embedding has constant precision, $\mathbf{B}$, while in the case of heavy-tailed noise, the precision is modulated by the data-dependent factor, $\frac{\nu'}{\nu+E_j}$.

Let us examine E_j in more detail, which is very interesting indeed. It turns out to be an *ancillary statistic*—it is *independent* of $\mathbf{z}$, because $\mathbf{GF} = \mathbf{0}$

[1] *Precision* for the t-distribution is not quite the same as inverse variance. The variance of a t-distribution is defined only for $\nu' > 2$ and is given in this case by $\frac{\nu+E_j}{\nu'-2}\,\mathbf{B}^{-1}$.

and so:

$$\mathbf{Gr} = \mathbf{G}(\mathbf{Fz} + \boldsymbol{\eta}) = \mathbf{G}\boldsymbol{\eta} \tag{7.10}$$

Even though E_j is independent of $\mathbf{z}$, which we are trying to infer, it is nevertheless relevant to that inference, because it modulates the uncertainty. Notice that this statistic only works for $D > d$. If we have $d = D$, and $\mathbf{F}$ is invertible, then $\mathbf{G} = \mathbf{0}$ and $E_j = 0$.

To better understand what $\mathbf{r}'\mathbf{Gr}$ does, consider applying an invertible linear transform to the data, such that the model for the transformed data then has $\mathbf{W} = \mathbf{I}$. Since this transform does not change anything essential, the role played by $\mathbf{G}$ is much the same as in the general case. For $\mathbf{W} = \mathbf{I}$ we have:

$$\mathbf{G} = \mathbf{I} - \mathbf{F}(\mathbf{F}'\mathbf{F})^{-1}\mathbf{F}$$

which is an *idempotent projection matrix*, for which:

$$E_j = \mathbf{r}_j'\mathbf{G}\mathbf{r}_j = (\mathbf{Fz} + \boldsymbol{\eta}_j)'\mathbf{G}(\mathbf{Fz} + \boldsymbol{\eta}_j) = \boldsymbol{\eta}_j'\mathbf{G}\boldsymbol{\eta}_j = (\mathbf{G}\boldsymbol{\eta}_j)'(\mathbf{G}\boldsymbol{\eta}_j)$$

That is, $\mathbf{Gr}_j$ is the projection of $\mathbf{r}_j$ onto the complement of the subspace spanned by the columns of $\mathbf{F}$. Since $\mathbf{Gr}_j$ is $(D-d)$-dimensional, $(\mathbf{Gr}_j)'(\mathbf{Gr}_j)$ is effectively composed of the sum of $D - d$ squares. Intuitively,

$$\beta_j = \frac{\nu'}{\nu + E_j} = \frac{\nu + (D - d)}{\nu + E_j}$$

forms a regularized estimate of the hidden noise precision modulation factor $\frac{\nu}{\lambda_j}$. The scale of E_j is proportional to $D - d$ and will (as indeed intuition suggests) have a greater effect for larger dimensionality differences. The scale of E_j is however independent of ν, so that for Gaussian noise, where $\nu \to \infty$ and $\frac{\nu}{\lambda_j} \to 1$, we also have $\frac{\nu'}{\nu + E_j} \to 1$.

7.1.4 Generalizations

Our model can be generalized in the following ways:

- Our derivation of the t-distribution, with precision scaling factor drawn from χ^2_ν is from Wikipedia.[2] Noting that $\chi^2_\nu(\lambda) = \mathcal{G}(\lambda \mid \frac{\nu}{2}, \frac{1}{2})$, where $\mathcal{G}$ is the gamma distribution, we can indeed generalize the model slightly to non-integer degrees of freedom, by drawing λ from a Gamma distribution—see Bishop's equation (2.161) in chapter 2 of [33].

[2] http://en.wikipedia.org/wiki/Multivariate_t-distribution.

- Consider making the *whole noise precision matrix* hidden, rather than just a scalar precision modulation factor. That would require a Wishart prior, rather a Gamma prior. Surprisingly, this does *not* lead to a generalization. When this matrix is marginalized out, we get exactly the *same* t-distribution likelihood. Compare for example Bishop's equations (2.161) and (B.68).

7.1.5 LR computation

LRs between TMEs are not available in closed form. However, as we found above, when $(D - d) \gg 1$, the TME is very close to a GME, even though the noise may be very heavy tailed, with a small ν. Let us proceed with the Gaussian approximation. As we already know, GME LRs are tractable, but slower than quadratic GPLDA scoring. Fortunately, if precision matrices differ only by a scaling factor, we can find faster recipes than for general GMEs. Below we consider an exact solution and a Taylor series approximation.

Exact GME scoring

We are interested in matrix factorizations of matrices of the form $\mathbf{I} + \beta\mathbf{B}$, where β varies but $\mathbf{B}$ remains fixed. It turns out we do not have to redo the matrix factorization for every β. Since $\mathbf{B}$ is symmetric positive semi-definite, we can factorize it as:

$$\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}' \tag{7.11}$$

where $\mathbf{V}$ is an orthonormal matrix having the eigenvectors along its columns, while $\mathbf{\Lambda}$ is a diagonal matrix of real, non-negative eigenvalues. By orthonormality we have $\mathbf{I} = \mathbf{V}\mathbf{V}'$, so that:

$$\mathbf{I} + \beta\mathbf{B} = \mathbf{V}(\mathbf{I} + \beta\mathbf{\Lambda})\mathbf{V}' \tag{7.12}$$

for any $\beta \geq 0$, the matrix of interest has a factorization of the same form, where we can re-use $\mathbf{V}$. In short, the eigenvectors remain fixed, while if λ is an eigenvalue of $\mathbf{B}$, then the corresponding eigenvalue of $\mathbf{I} + \beta\mathbf{B}$ is $1 + \beta\lambda$.

One way to structure the calculations is to transform the hidden variable as $\tilde{\mathbf{z}} = \mathbf{V}'\mathbf{z}$. The Jacobian determinant magnitude of this transform is unity, it leaves the standard normal prior unchanged and it diagonalizes all precision matrices.

Taylor series approximation

We examine the Taylor series approximation for this case in more detail.

Let $f_j(\mathbf{z}) = \exp[\mathbf{z}'\mathbf{a}_j, \beta_j\mathbf{B}]$. Pooling is done by simple addition of the $\mathbf{a}$ and β parameters and gives a representation in the same form. We can therefore do any LR computation by using identical building blocks, of the form

$$F(\mathbf{a}_i, \mathbf{a}_j, \beta_i, \beta_j) = \log\langle \overline{f}_i, \overline{f}_j \rangle \tag{7.13}$$

Let us approximate this using a variant of the symmetrical Taylor series approximation about some fixed value $\beta_0 > 0$. Letting $\delta_i = \beta_i - \beta_0$ and $\delta_j = \beta_j - \beta$) and $F_{ij} = F(\mathbf{a}_i, \mathbf{a}_j, \beta_0, \beta_0)$ we do:

$$F(\mathbf{a}_i, \mathbf{a}_j, \beta_i, \beta_j) \approx F_{ij} + \frac{\partial F_{ij}}{\partial \beta}(\delta_i + \delta_j) + \frac{1}{2}\begin{bmatrix}\delta_i & \delta_j\end{bmatrix}\mathbf{H}\begin{bmatrix}\delta_i \\ \delta_j\end{bmatrix} \tag{7.14}$$

where by symmetry $\frac{\partial F_{ij}}{\partial \beta} = \frac{\partial F_{ij}}{\partial \beta_i} = \frac{\partial F_{ij}}{\partial \beta_j}$. The Hessian, H, is fixed, because it depends only on β_0, but fortunately not on $(\mathbf{a}_i, \mathbf{a}_j)$.

Other variants are possible, for example we could let $\gamma = \log(\beta + c)$ and do the expansion in terms of γ instead. In preliminary experiments, we found that choosing c as the reciprocal of the mean of the eigenvalues of $\mathbf{B}$ gives an almost linear relationship between γ and $\log|\mathbf{I} + \beta\mathbf{B}|$.

7.1.6 Multiclass classification

For multiclass classification, with large amounts of training data for each class, we can apply the scoring approximation of section 6.5 to this model. We assume that the normalized distribution (product of t-distributions) that represents the training data for every class i, concentrates to a Dirac delta at $\hat{\mathbf{z}}_i$. As explained above, for runtime scoring, we can simply plug these values into the meta-embeddings that represent the test recordings.

Training

Let class i have a set of training data, $\mathcal{R}_i = \{\mathbf{r}_{ij}\}_{j=1}^{N_i}$, where each $\mathbf{r}_{ij}$ is represented by (7.9) as the approximate Gaussian meta-embedding:

$$f_{ij}(\mathbf{z}) = \mathcal{N}\big(\mathbf{z} \mid \boldsymbol{\mu}_{ij}, (\beta_{ij}\mathbf{B})^{-1}\big) \tag{7.15}$$

where by (7.8):

$$\boldsymbol{\mu}_{ij} = \mathbf{B}^{-1}\mathbf{F}'\mathbf{W}\mathbf{r}_{ij}\,, \qquad \beta_{ij} = \frac{\nu + D - d}{\nu + E_{ij}} \qquad \text{and} \qquad E_{ij} = \mathbf{r}_{ij}'\mathbf{G}\mathbf{r}_{ij} \tag{7.16}$$

This gives:

$$\pi(\mathbf{z})f_i(\mathbf{z}) = \pi(\mathbf{z})\prod_j f_{ij}(\mathbf{z}) \propto P(\mathbf{z} \mid \mathcal{R}_i, \pi) = \mathcal{N}(\mathbf{z} \mid \hat{\mathbf{z}}_i, \bar{\mathbf{B}}_i^{-1}) \tag{7.17}$$

where

$$\bar{\mathbf{B}}_i = \mathbf{I} + \sum_j \beta_{ij}\mathbf{B} \tag{7.18}$$

and

$$\hat{\mathbf{z}}_i = \bar{\mathbf{B}}_i^{-1}\mathbf{B}\sum_j \beta_{ij}\boldsymbol{\mu}_{ij} = \bar{\mathbf{B}}_i^{-1}\mathbf{B}\sum_j \beta_j\boldsymbol{\mu}_{ij} = \bar{\mathbf{B}}^{-1}\mathbf{B}\mathbf{F}'\mathbf{W}\sum_j \beta_j\mathbf{r}_{ij} \tag{7.19}$$

Notice that the class representative, $\hat{\mathbf{z}}_i$, is formed via a weighted combination of the data for that class. The weights are inversely proportional to the estimated noise variances. If we have enough data, so that $\bar{\mathbf{B}}_i \gg \mathbf{I}$, then we can approximate:

$$\pi(\mathbf{z})\overline{f_i} \approx \delta(\mathbf{z} - \hat{\mathbf{z}}_i)$$

Scoring

At runtime, given a test recording, r_j, represented by the approximate Gaussian meta-embedding:

$$g_j(\mathbf{z}) = \exp[\mathbf{a}_j'\mathbf{z} - \frac{\beta_j}{2}\mathbf{z}'\mathbf{B}\mathbf{z}] \tag{7.20}$$

we can use (6.18) to approximate the log-likelihood for each class i as:

$$\begin{aligned}\log L_{ij} &\approx \log g_j(\hat{\mathbf{z}}_i) - \log\langle g_j\rangle \\ &= \mathbf{a}_j'\mathbf{z}_i - \frac{\beta_j}{2}\hat{\mathbf{z}}_i'\mathbf{B}\hat{\mathbf{z}}_i - \frac{1}{2}\mathbf{a}_j(\mathbf{I} + \beta_j\mathbf{B})^{-1}\mathbf{a}_j - \frac{1}{2}\log|\mathbf{B}| - \frac{d}{2}\log\beta_j\end{aligned} \tag{7.21}$$

For the open-set case, we can omit the constant $\frac{1}{2}\log|\mathbf{B}|$, while for the *closed-set* case, we can omit all terms independent of i, to give:

$$\log L_{ij} \approx \mathbf{a}_j'\hat{\mathbf{z}}_i - \frac{\beta_j}{2}\hat{\mathbf{z}}_i'\mathbf{B}\hat{\mathbf{z}}_i + \text{const} \tag{7.22}$$

This can be arranged into a $(d+1)$-dimensional dot-product for computational convenience.

Intuitively it would be satisfying for the the score to be modulated by β_j, so that score magnitudes become smaller for noisier data. This is true for

the second term in (7.22), but what about the first term? We can make the the dependence on β_j clear by considering $\boldsymbol{\mu}_j = (\mathbf{I} + \beta_j \mathbf{B})^{-1}\mathbf{a}_j$, which is the mean of the Gaussian approximation to $P(\mathbf{z} \mid \mathbf{r}_j, \pi)$. Recall that similarly, $\hat{\mathbf{z}}_i$ is the mean of the Gaussian approximation to $P(\mathbf{z} \mid \mathcal{R}_i, \pi)$. The first term can now be rewritten as:

$$\mathbf{a}_j' \hat{\mathbf{z}}_i = \boldsymbol{\mu}_i'(\mathbf{I} + \beta_j \mathbf{B})\hat{\mathbf{z}}_i$$

where the role of β_j becomes clear.

As a sanity check, let's derive this scoring recipe directly from (5.6):

$$\lim_{\beta_i \to \infty} \log E(\mathbf{a}_i + \mathbf{a}_j, (\beta_i + \beta_j)\mathbf{B}) - \log E(\mathbf{a}_j, \beta_i \mathbf{B}) \tag{7.23}$$

Doing it without Gaussians

As an alternative (perhaps a good idea when $D - d$ is small), both training and scoring can be done without Gaussian approximations. For training, the class representatives, $\hat{\mathbf{z}}_i$, can be found by numerical maximization of the product of t-distributions. If in doubt about the sharpness of the peaks at the maxima, we could verify this by using the Laplace approximation as explained in section 6.4. If all peaks are indeed sharp, we can score closed-set multiclass likelihoods by plugging the $\hat{\mathbf{z}}_i$ into the t-distribution meta-embeddings of the test segments, as in (6.18).

7.2 Mixture of PLDAs

Our PLDA model with heavy-tailed noise makes use of a continuous mixture of Gaussians. The resultant meta-embeddings are t-distributions, which we then resorted to approximate by discrete mixtures of Gaussians. In this section we investigate instead what happens if we start out with a discrete mixture of PLDA models instead.

In this generative model, the prior remains standard normal, but the likelihood is now a mixture of the form:

$$f_j(\mathbf{z}) \propto P(\mathbf{r}_j \mid \mathbf{z}) = \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{r}_j \mid \mathbf{F}_k \mathbf{z}, \mathbf{W}_k^{-1}) \tag{7.24}$$

We shall express the likelihood as a mixture of normalized GMEs of the form:

$$\overline{g_{jk}}(\mathbf{z}) = \frac{|\mathbf{I} + \mathbf{B}_k|^{\frac{1}{2}}}{\exp[\frac{1}{2}\mathbf{a}_{jk}'(\mathbf{I} + \mathbf{B}_k)\mathbf{a}_{jk}]} \exp[\mathbf{z}'\mathbf{a}_{jk} - \frac{1}{2}\mathbf{z}'\mathbf{B}_k\mathbf{z}] \tag{7.25}$$

with parameters of the form:

$$\mathbf{B}_k = \mathbf{F}_k' \mathbf{W}_k \mathbf{F}_k, \qquad \text{and} \qquad \mathbf{a}_{jk} = \mathbf{F}_k' \mathbf{W}_k \mathbf{r}_j \tag{7.26}$$

The likelihood can be rewritten without assuming invertibility of the $\mathbf{B}_k$ as:

$$\begin{aligned}
f_j(\mathbf{z}) &\propto \sum_{k=1}^{K} w_k \sqrt{|\mathbf{W}_k|} \exp\big[-\frac{1}{2}(\mathbf{r}_j - \mathbf{F}_k\mathbf{z})'\mathbf{W}_k(\mathbf{r}_j - \mathbf{F}_k\mathbf{z})\big] \\
&= \sum_{k=1}^{K} w_k \sqrt{|\mathbf{W}_k|} \exp\big[-\frac{1}{2}\mathbf{r}_j'\mathbf{W}_k\mathbf{r}_j - \frac{1}{2}\mathbf{z}'\mathbf{B}_k\mathbf{z} + \mathbf{z}'\mathbf{a}_{jk})\big] \\
&= \sum_{k=1}^{K} w_k \sqrt{\frac{|\mathbf{W}_k|}{|\mathbf{I}+\mathbf{B}_k|}} \exp\big[\frac{1}{2}\mathbf{a}_{jk}'(\mathbf{I}+\mathbf{B}_k)^{-1}\mathbf{a}_{jk} - \frac{1}{2}\mathbf{r}_j'\mathbf{W}_k\mathbf{r}_j\big]\, \overline{g_{jk}}(\mathbf{z}) \\
&= \sum_{k=1}^{K} w_k \sqrt{\frac{|\mathbf{W}_k|}{|\mathbf{I}+\mathbf{B}_k|}} \exp\big[\frac{1}{2}\mathbf{r}_j'(\mathbf{W}_k\mathbf{F}_k(\mathbf{I}+\mathbf{B}_k)^{-1}\mathbf{F}_k'\mathbf{W}_k - \mathbf{W}_k)\mathbf{r}_j\big]\, \overline{g_{jk}}(\mathbf{z}) \\
&= \sum_{k=1}^{K} w_k \sqrt{\frac{|\mathbf{W}_k|}{|\mathbf{I}+\mathbf{B}_k|}} \exp\big[-\frac{1}{2}\mathbf{r}_j'\tilde{\mathbf{G}}_k\mathbf{r}_j\big]\, \overline{g_{jk}}(\mathbf{z})
\end{aligned} \tag{7.27}$$

where $\tilde{\mathbf{G}}_k$ is defined below and simplified with the Woodbury identity:[3]

$$\tilde{\mathbf{G}}_k = \mathbf{W}_k - \mathbf{W}_k\mathbf{F}_k(\mathbf{I}+\mathbf{B}_k)^{-1}\mathbf{F}_k'\mathbf{W}_k = (\mathbf{W}^{-1} + \mathbf{F}\mathbf{F}')^{-1} \tag{7.28}$$

The RHS shows that $\tilde{\mathbf{G}}_k^{-1}$ is the variance of $\mathbf{r}$, when sampled from:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \qquad \text{and} \qquad \mathbf{r} \sim \mathcal{N}(\mathbf{F}_k\mathbf{z}, \mathbf{W}_k^{-1})$$

The matrix determinant lemma[4] further reveals that:

$$|\tilde{\mathbf{G}}_k| = \frac{|\mathbf{W}_k|}{|\mathbf{I}+\mathbf{B}_k|} \tag{7.29}$$

so that we can finally simplify the meta-embedding as:

$$\begin{aligned}
f_j(\mathbf{z}) &\propto \sum_{k=1}^{K} w_k\, \mathcal{N}(\mathbf{r}_j \mid \mathbf{0}, \tilde{\mathbf{G}}_k^{-1})\, \overline{g_{jk}}(\mathbf{z}) \\
&\propto \sum_{k=1}^{K} P(k \mid \mathbf{r}_j)\, \overline{g_{jk}}(\mathbf{z})
\end{aligned} \tag{7.30}$$

[3] http://en.wikipedia.org/wiki/Woodbury_matrix_identity
[4] http://en.wikipedia.org/wiki/Matrix_determinant_lemma

Chapter 8

Discriminative meta-embeddings

8.1 Pairs

8.2 Triplet loss

First pubished in [2].

Good paper: [29]: explains original triplet loss, variants for hinge vs softplus loss and gives nice recipe for mining moderately hard triplets,

8.3 Multiclass classification

[8, 9]

8.4 Pseudolikelihood

Appendix A

Multidimensional Fourier transform

In a mathematically rigorous treatment of Fourier transforms, one has to carefully define the function spaces in which the transforms can be applied [36]. In this document, we shall mostly just tacitly assume that wherever we employ definite integrals, that those integrals exist and are finite.

The *Fourier transform* (FT) is an *operator*—it maps one function to another function. We are interested in the n-dimensional FT, which maps some function, $f : \mathbb{R}^d \to \mathbb{C}$ to its transform, $\tilde{f} : \mathbb{R}^d \to \mathbb{C}$, where $\mathbb{R}^d$ is n-dimensional real Euclidean space and $\mathbb{C}$ is the field of complex numbers. Denoting the transform by $\tilde{f} = \mathcal{F}\{f\}$ and letting $\mathbf{z}, \boldsymbol{\zeta} \in \mathbb{R}^d$, we have:

$$\mathcal{F}\{f\}(\boldsymbol{\zeta}) = \tilde{f}(\boldsymbol{\zeta}) = \int_{\mathbb{R}^d} f(\mathbf{z}) e^{-2\pi i \mathbf{z}'\boldsymbol{\zeta}} \, d\mathbf{z} \tag{A.1}$$

where $\mathbf{z}'\boldsymbol{\zeta}$ denotes dot product. The inverse transform (IFT) is:

$$\mathcal{F}^{-1}\{\tilde{f}\}(\mathbf{z}) = f(\mathbf{z}) = \int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta}) e^{2\pi i \mathbf{z}'\boldsymbol{\zeta}} \, d\boldsymbol{\zeta} \tag{A.2}$$

Below we shall sometimes use the notation $\mathcal{F}_{\mathbf{z}}\{f(\mathbf{z}, \mathbf{y})\}$ to denote that the transform is done w.r.t. $\mathbf{z}$.

A.1 Properties

A.1.1 Symmetry

A real-valued function, $k : \mathbb{R}^d \to R$, which is also symmetric, $k(\mathbf{z}) = k(-\mathbf{z})$, has a real-valued FT: $\tilde{k}(\boldsymbol{\zeta}) = \overline{\tilde{k}(\boldsymbol{\zeta})}$, where the overline indicates complex conjugation. This follows readily from the definition (A.1).

A.1.2 The convolution theorem

Given functions f and g, their *convolution* is defined as:

$$(f * g)(\mathbf{z}) = \int_{\mathbb{R}^d} f(\mathbf{z}-\mathbf{y})g(\mathbf{y})\,d\mathbf{y} = \int_{\mathbb{R}^d} f(\mathbf{y})g(\mathbf{z}-\mathbf{y})\,d\mathbf{y} \tag{A.3}$$

which simplifies to multiplication in the transform domain:

$$\mathcal{F}\{f * g\}(\boldsymbol{\zeta}) = \tilde{f}(\boldsymbol{\zeta})\tilde{g}(\boldsymbol{\zeta}) \tag{A.4}$$

where $\tilde{f} = \mathcal{F}\{f\}$ and $\tilde{g} = \mathcal{F}\{g\}$.

A.1.3 Dirac delta

The *dirac Delta*, δ, has the property that:

$$\int_{\mathbb{R}^d} \delta(\mathbf{z}-\mathbf{y})f(\mathbf{z})\,d\mathbf{z} = f(\mathbf{y}) \tag{A.5}$$

which can be used to compute its FT as:

$$\mathcal{F}_{\mathbf{z}}\{\delta(\mathbf{z}-\mathbf{y})\}(\boldsymbol{\zeta}) = \int_{\mathbb{R}^d} \delta(\mathbf{z}-\mathbf{y})e^{-2\pi\mathbf{z}'\boldsymbol{\zeta}}\,d\mathbf{z} = e^{-2\pi i\mathbf{y}'\boldsymbol{\zeta}} \tag{A.6}$$

where we have introduced a subscript for the operator, $\mathcal{F}_{\mathbf{z}}$, to make clear that it operates on $\delta(\mathbf{z}-\mathbf{y})$ as a function of $\mathbf{z}$. Applying the IFT, $\mathcal{F}_{\boldsymbol{\zeta}}^{-1}$, on both sides of (A.6) gives:

$$\delta(\mathbf{z}-\mathbf{y}) = \int_{\mathbb{R}^d} e^{-2\pi i\mathbf{y}'\boldsymbol{\zeta}}e^{2\pi i\mathbf{z}'\boldsymbol{\zeta}}\,d\boldsymbol{\zeta} = \int_{\mathbb{R}^d} e^{2\pi i(\mathbf{z}-\mathbf{y})'\boldsymbol{\zeta}}\,d\boldsymbol{\zeta} \tag{A.7}$$

A.1.4 Shifting property

If $\mathcal{F}_{\mathbf{z}}\{f(\mathbf{z})\}(\boldsymbol{\zeta}) = \tilde{f}(\boldsymbol{\zeta})$, we want to express the FT of a shifted version of f, namely $f(\mathbf{z}-\mathbf{y})$ in terms of $\tilde{f}$. Letting $\mathbf{r} = \mathbf{z}-\mathbf{y}$, and noting that $d\mathbf{r} = d\mathbf{z}$, we have:

$$\begin{aligned}
\mathcal{F}_{\mathbf{z}}\{f(\mathbf{z}-\mathbf{y})\}(\boldsymbol{\zeta}) &= \int_{\mathbb{R}^d} f(\mathbf{z}-\mathbf{y})e^{-2\pi i\mathbf{z}'\boldsymbol{\zeta}}\,d\mathbf{z} \\
&= \int_{\mathbb{R}^d} f(\mathbf{r})e^{-2\pi i(\mathbf{r}+\mathbf{y})'\boldsymbol{\zeta}}\,d\mathbf{r} \\
&= e^{-2\pi i\mathbf{y}'\boldsymbol{\zeta}}\int_{\mathbb{R}^d} f(\mathbf{r})e^{-2\pi i\mathbf{r}'\boldsymbol{\zeta}}\,d\mathbf{r} \\
&= e^{-2\pi i\mathbf{y}'\boldsymbol{\zeta}}\tilde{f}(\boldsymbol{\zeta})
\end{aligned} \tag{A.8}$$

Shifting in the primary domain causes amplitude modulation in the transformed domain.

A.1.5 Linear transformation property

Let $\mathbf{y} = \mathbf{A}\mathbf{z}$, where $\mathbf{A}$ is an invertible square matrix. Then we have that $\mathbf{z} = \mathbf{A}^{-1}\mathbf{y}$ and $d\mathbf{y} = |\det(\mathbf{A})|\, d\mathbf{z}$.

$$\begin{aligned}\mathcal{F}\{f(\mathbf{A}\mathbf{z})\}(\boldsymbol{\zeta}) &= \int_{\mathbb{R}^d} f(\mathbf{A}\mathbf{z}) e^{-2\pi i \mathbf{z}'\boldsymbol{\zeta}}\, d\mathbf{z} \\ &= \frac{1}{|\det(\mathbf{A})|} \int_{\mathbb{R}^d} f(\mathbf{y}) e^{-2\pi i (\mathbf{A}^{-1}\mathbf{y})'\boldsymbol{\zeta}}\, d\mathbf{y} \\ &= \frac{1}{|\det(\mathbf{A})|} \int_{\mathbb{R}^d} f(\mathbf{y}) e^{-2\pi i \mathbf{y}'(\mathbf{A}^{-1})'\boldsymbol{\zeta}}\, d\mathbf{y} \\ &= \frac{1}{|\det(\mathbf{A})|} \mathcal{F}\{f\}\big((\mathbf{A}^{-1})'\boldsymbol{\zeta}\big)\end{aligned} \tag{A.9}$$

A.1.6 Definite integral

It follows directly from the definition (A.1) that:

$$\int_{\mathbb{R}^d} f(\mathbf{z})\, d\mathbf{z} = \int_{\mathbb{R}^d} f(\mathbf{z}) e^0\, d\mathbf{z} = \mathcal{F}\{f\}(0) \tag{A.10}$$

A.1.7 Parseval / Plancherel

Here we need to assume that f, g and their transforms, $\tilde{f}, \tilde{g}$ live in a Hilbert function space, with inner product defined as:

$$\langle f, g\rangle = \int_{\mathbb{R}^d} f(\mathbf{z})\overline{g(\mathbf{z})}\, d\mathbf{z} = \overline{\int_{\mathbb{R}^d} g(\mathbf{z})\overline{f(\mathbf{z})}\, d\mathbf{z}} = \overline{\langle g, f\rangle} \tag{A.11}$$

where the overline denotes complex conjugate. This Hilbert space is also the Lp function space, $L^2(\mathbb{R}^d)$, for which $\|f\|_2, \|g\|_2 \leq \infty$, where we use the Hilbert space norm $\|f\|_2 = \sqrt{\langle f, f\rangle}$.

Note the distinction between norm, $\|f\|_2$, which is a scalar and absolute value, $|f(\mathbf{z})|$ which is a function of $\mathbf{z}$. For a complex-valued function, the absolute value can be defined as as:

$$|f(\mathbf{z})| = \sqrt{f(\mathbf{z})\overline{f(\mathbf{z})}} \tag{A.12}$$

Of course, we have: $\|f\|_2^2 = \int_{\mathbb{R}^d} |f(\mathbf{z})|^2\, d\mathbf{z}$.

The Parseval / Plancherel[1] theorem equates the original inner-product to one between their Fourier transforms: $\langle f, g\rangle = \langle \tilde{f}, \tilde{g}\rangle$, where the RHS may

[1] One theorem says $\langle f, f\rangle = \langle \tilde{f}, \tilde{f}\rangle$ and the other, more generally, $\langle f, g\rangle = \langle \tilde{f}, \tilde{g}\rangle$, but I cannot figure out from the literature which one is Plancherel and which Parseval.

sometimes be more convenient to evaluate. More specifically, we also have $\|f\|_2 = \|\tilde{f}\|_2$. To show this, we expand the inner-product in terms of the inverse transforms of $\tilde{f}$ and $\tilde{g}$, rearrange, employ (A.7) and then (A.5):

$$\begin{aligned}
\langle f, g \rangle &= \int_{\mathbb{R}^d} f(\mathbf{z})\overline{g(\mathbf{z})}\, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} \mathcal{F}^{-1}\{\tilde{f}\}(\mathbf{z})\overline{\mathcal{F}^{-1}\{\tilde{g}\}(\mathbf{z})}\, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} \left(\int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta}) e^{2\pi i \boldsymbol{\zeta}'\mathbf{z}}\, d\boldsymbol{\zeta} \right) \overline{\left(\int_{\mathbb{R}^d} \tilde{g}(\mathbf{y}) e^{2\pi i \mathbf{y}'\mathbf{z}}\, d\mathbf{y} \right)}\, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} \left(\int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta}) e^{2\pi i \boldsymbol{\zeta}'\mathbf{z}}\, d\boldsymbol{\zeta} \right) \left(\int_{\mathbb{R}^d} \overline{\tilde{g}(\mathbf{y})} e^{-2\pi i \mathbf{y}'\mathbf{z}}\, d\mathbf{y} \right)\, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta})\overline{\tilde{g}(\mathbf{y})} \left(\int_{\mathbb{R}^d} e^{2\pi i (\boldsymbol{\zeta}-\mathbf{y})'\mathbf{z}}\, d\mathbf{z} \right)\, d\boldsymbol{\zeta}\, d\mathbf{y} \\
&= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta})\overline{\tilde{g}(\mathbf{y})} \delta(\boldsymbol{\zeta} - \mathbf{y})\, d\boldsymbol{\zeta}\, d\mathbf{y} \\
&= \int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta})\overline{\tilde{g}(\boldsymbol{\zeta})}\, d\boldsymbol{\zeta} \\
&= \langle \tilde{f}, \tilde{g} \rangle
\end{aligned} \tag{A.13}$$

This means we can form the dot product either in the original domain, or in the transformed domain.

A.2 Characteristic function

The characteristic function of a probability distribution, P, over $\mathbf{z} \in \mathbb{R}^d$ is usually defined as, $\phi_P : \mathbb{R}^d \to \mathbb{C}$, where:

$$\phi_P(\mathbf{t}) = \left\langle e^{i\mathbf{t}'\mathbf{z}} \right\rangle_{\mathbf{z}\sim P} \tag{A.14}$$

Notice that since $|e^{i\mathbf{t}'\mathbf{z}}| \leq 1$, we also have $|\phi_P(\mathbf{t})| \leq 1$. Indeed, the characteristic function always exists. If P has a density function, $p(\mathbf{z})$, then density and characteristic function essentially form a Fourier transform pair:

$$\phi_p(\mathbf{t}) = \mathcal{C}\{p\}(\mathbf{t}) = \int_{\mathbb{R}^d} e^{i\mathbf{t}'\mathbf{z}} p(\mathbf{z})\, d\mathbf{z} \tag{A.15}$$

where we denote this reparametrized version of the Fourier transform by the operator $\mathcal{C}$. Observe that if we let $\mathbf{t} = -2\pi\boldsymbol{\zeta}$, then:

$$\phi_p(-2\pi\boldsymbol{\zeta}) = \int_{\mathbb{R}^d} e^{-2\pi i \boldsymbol{\zeta}'\mathbf{z}} p(\mathbf{z})\, d\mathbf{z} = \tilde{p}(\boldsymbol{\zeta}) \tag{A.16}$$

or

$$\mathcal{C}\{p\}(-2\pi\boldsymbol{\zeta}) = \mathcal{F}\{p\}(\boldsymbol{\zeta}) \tag{A.17}$$

The characteristics functions for most standard probability distributions are known [37].

A.2.1 Parseval

How does Parseval's theorem translate in terms of characteristic functions? Let f, g be probability density functions with characteristic functions, ϕ_f and ϕ_g. Then, letting $\mathbf{t} = -2\pi\boldsymbol{\zeta}$ and $d\mathbf{t} = (2\pi)^d d\boldsymbol{\zeta}$

$$\begin{aligned}
\int_{\mathbb{R}^d} f(\mathbf{z})g(\mathbf{z})\, d\mathbf{z} &= \int_{\mathbb{R}^d} \tilde{f}(\boldsymbol{\zeta})\overline{\tilde{g}(\boldsymbol{\zeta})}\, d\boldsymbol{\zeta} \\
&= \int_{\mathbb{R}^d} \phi_f(-2\pi\boldsymbol{\zeta})\overline{\phi_g(-2\pi\boldsymbol{\zeta})}\, d\boldsymbol{\zeta} \\
&= \frac{1}{(2\pi)^d}\int_{\mathbb{R}^d} \phi_f(\mathbf{t})\overline{\phi_g(\mathbf{t})}\, d\mathbf{t}
\end{aligned} \tag{A.18}$$

A.2.2 Convolution

The convolution theorem also works for the characteristic function [37]:

$$\begin{aligned}
\mathcal{C}\{f * g\}(\mathbf{t}) &= \mathcal{F}\{f * g\}\big(\frac{-1}{2\pi}\mathbf{t}\big) \\
&= \mathcal{F}\{f\}\big(\frac{-1}{2\pi}\mathbf{t}\big)\, \mathcal{F}\{g\}\big(\frac{-1}{2\pi}\mathbf{t}\big) \\
&= \mathcal{C}\{f\}(\mathbf{t})\, \mathcal{C}\{g\}(\mathbf{t})
\end{aligned} \tag{A.19}$$

A.2.3 Gaussian characteristic function

The characteristic function for a multivariate Gaussian is [37]:

$$\begin{aligned}
\mathcal{C}\{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})\}(\mathbf{t}) &= \int_{\mathbb{R}^d} \mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \exp(i\mathbf{t}'\mathbf{z})\, d\mathbf{z} \\
&= \exp(i\boldsymbol{\mu}'\mathbf{t} - \frac{1}{2}\mathbf{t}'\boldsymbol{\Sigma}\mathbf{t}) \\
&= e^{i\boldsymbol{\mu}'\mathbf{t}}\sqrt{|2\pi\boldsymbol{\Sigma}^{-1}|}\,\mathcal{N}(\mathbf{t} \mid \mathbf{0}, \boldsymbol{\Sigma}^{-1})
\end{aligned} \tag{A.20}$$

Notice that for zero mean, $\boldsymbol{\mu} = \mathbf{0}$, the characteristic function is also an unnormalized zero mean Gaussian, with the roles of covariance and precision

interchanged. As the PDF gets more concentrated, its transform gets more spread out—this inverse scaling is of course due to (A.9). If $\boldsymbol{\mu}$ is non-zero, then the characteristic function is not even real-valued and is therefore not a probability density.

A.2.4 Empirical characteristic function

The empirical distribution of m observed data points, $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m \in \mathbb{R}^d$, does not have a density function in the strict sense, but the following mixture of Dirac delta's can be used for most purposes as a drop-in replacement for the density:

$$p(\mathbf{z}) = \frac{1}{m} \sum_{\ell=1}^{m} \delta(\mathbf{z} - \mathbf{z}_\ell) \tag{A.21}$$

The characteristic function, using (A.6), is:

$$\tilde{p}(\boldsymbol{\zeta}) = \frac{1}{m} \sum_{\ell=1}^{m} e^{-2\pi i \boldsymbol{\zeta}' \mathbf{z}_\ell} \tag{A.22}$$

A.2.5 Positive definite function

We term a function $k : \mathbb{R}^d \to \mathbb{C}$ *positive definite*, if for every $m \geq 1$ and every $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m \in \mathbb{R}^d$ and every $\alpha_1, \alpha_2, \ldots, \alpha_m \in \mathbb{C}$, we have:

$$\sum_{\ell=1}^{m} \sum_{j=1}^{m} \alpha_\ell \overline{\alpha_j} k(\mathbf{z}_\ell - \mathbf{z}_j) \geq 0 \tag{A.23}$$

This is the same as requiring that any m-by-m Gram matrix formed with elements $k(\mathbf{z}_i - \mathbf{z}_j)$ must be Hermitian and positive (semi) definite.

Notice that for positive definite k, we can define a shift-invariant *kernel function*, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{C}$, as $K(\mathbf{z}, \mathbf{y}) = k(\mathbf{z} - \mathbf{y})$, so that K is positive definite in the sense required for the theory of reproducing kernel Hilbert spaces (RKHS) and kernel methods in machine learning (SVM, Gaussian processes, etc.).

A.2.6 Bochner's theorem

Let $r(\mathbf{z})$, be a *probability density function* on $\mathbf{z} \in \mathbb{R}^d$. We show that its characteristic function, $\tilde{r}$, is positive definite:

$$\begin{aligned}
&\sum_{\ell=1}^{m}\sum_{j=1}^{m} \alpha_\ell \overline{\alpha_j} \tilde{r}(\boldsymbol{\zeta}_\ell - \boldsymbol{\zeta}_j) \\
&= \sum_{\ell=1}^{m}\sum_{j=1}^{m} \alpha_\ell \overline{\alpha_j} \int_{\mathbb{R}^d} r(\mathbf{z}) e^{-2\pi i (\boldsymbol{\zeta}_\ell - \boldsymbol{\zeta}_j)' \mathbf{z}} \, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} r(\mathbf{z}) \sum_{\ell=1}^{m}\sum_{j=1}^{m} \alpha_\ell e^{-2\pi i \boldsymbol{\zeta}_\ell' \mathbf{z}} \overline{\alpha_j e^{-2\pi i \boldsymbol{\zeta}_j' \mathbf{z}}} \, d\mathbf{z} \\
&= \int_{\mathbb{R}^d} r(\mathbf{z}) \left| \sum_{\ell=1}^{m} \alpha_\ell e^{-2\pi i \boldsymbol{\zeta}_\ell' \mathbf{z}} \right|^2 d\mathbf{z} \geq 0
\end{aligned} \tag{A.24}$$

Since $r(\mathbf{z})$ is normalized, we also have by (A.10), that $\tilde{r}(\mathbf{0}) = 1$. Bochner's theorem says that the converse (although harder to prove) is also true—the FT (or IFT) of a positive definite function (which evaluates to 1 at $\mathbf{0}$), is a probability density [36].

Bibliography

[1] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *NIPS*, 2000.

[2] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832

[3] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Interspeech*, Brighton, UK, September 2009.

[4] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[5] D. Martínez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in ivectors space," in *Interspeech*, 2011.

[6] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Interspeech*, 2017. [Online]. Available: https://arxiv.org/abs/1706.08612

[7] D. Snyder, P. Ghahremani, and D. Povey, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *IEEE Workshop on Spoken Language Technology*, 2016.

[8] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *Interspeech*, Stockholm, 2017.

[9] G. Gelly and J. L. Gauvain, "Spoken language identification using lstm-based angular proximity," in *Interspeech*, Stockholm, 2017.

[10] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," CRIM, Montreal, Tech. Rep. CRIM-06/08-13, 2005.

[11] ——, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey Speaker and Language Recognition Workshop*, Brno, Czech Republic, June 2010, keynote presentation.

[12] S. Ioffe, "Probabilistic linear discriminant analysis," in *9th European Conference on Computer Vision*, Graz, Austria, 2006.

[13] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th International Conference on Computer Vision*, 2007.

[14] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. Prince, "Probabilistic linear discriminant analysis for inferences about identity," *IEEE Trans. PAMI*, vol. 34, no. 1, January 2012.

[15] L. Vilnis and A. McCallum, "Word representations via Gaussian embedding," in *ICLR*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6623

[16] S. Cumani, O. Plchot, and P. Laface, "On the use of i-vector posterior distributions in PLDA," *IEEE Trans. ASLP*, vol. 22, no. 4, 2014.

[17] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam, and P. Dumouchel, "PLDA for speaker verification with utterances of arbitrary duration," in *IEEE ICASSP*, 2013.

[18] T. Stafylakis, P. Kenny, P. Ouellet, J. Perez, M. Kockmann, and P. Dumouchel, "Text-dependent speaker recognition using PLDA with uncertainty propagation," in *Interspeech*, 2013.

[19] P. Kenny, T. Stafylakis, J. Alam, V. Gupta, and M. Kockmann, "Uncertainty modeling without subspace methods for text-dependent speaker recognition," in *Speaker Odyssey: The Speaker and Language Recognition Workshop*, Bilbao, 2016.

[20] Y. S. Chow and H. Teicher, *Probability theory: Independence, interchangeability, martingales*, 3rd ed., ser. Springer Texts in Statistics. New York: Springer, 1997.

[21] N. Brümmer and J. du Preez, "Application independent evaluation of speaker detection," *Computer Speech and Language*, vol. 20, no. 2-3, pp. 230–275, 2006.

[22] N. Brümmer and E. de Villiers, "The speaker partitioning problem," in *Odyssey Speaker and Language Recognition Workshop*, Brno, Czech Republic, June 2010.

[23] E. T. Jaynes, *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

[24] K. Siegrist, "Random: Probability, mathematical statistics, stochastic processes," 1997, see section 3.11 Vector Spaces of Random Variables. [Online]. Available: http://www.math.uah.edu/stat/expect/Spaces.html

[25] P. Billingsley, *Probability and Measure*, 3rd ed. John Wiley & Sons, 1995.

[26] P. Ouwehand, "Spaces of random variables," AIMS, lecture, November 2010. [Online]. Available: http://users.aims.ac.za/~pouw/Lectures/Lecture_Spaces_Random_Variables.pdf

[27] D. Bigoni, "Uncertainty quantification with applications to engineering problems," Ph.D. dissertation, Thechnical University of Denmark, 2015, see Appendix B: Probability theory and functional spaces.

[28] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Interspeech*, Florence, Italy, 2011.

[29] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *CoRR*, vol. abs/1703.07737, 2017. [Online]. Available: http://arxiv.org/abs/1703.07737

[30] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *arXiv*, 2013. [Online]. Available: http://arxiv.org/abs/1312.6114

[31] M. Titsias and M. Lázaro-Gredilla, "Doubly stochastic variational Bayes for non-conjugate inference," in *ICML*, 2014. [Online]. Available: www.jmlr.org/proceedings/papers/v32/titsias14.pdf

[32] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[33] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[34] A. F. Martin, C. S. Greenberg, J. M. Howard, G. R. Doddington, and J. J. Godfrey, "NIST language recognition evaluation: Past and future," in *Speaker Odyssey: The Speaker and Language Recognition Workshop*, Joensuu, Finland, 2014.

[35] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey 2010, The Speaker and Language Recognition Workshop, Brno*, 2010.

[36] E. M. Stein and G. Weiss, *Introduction to Fourier Analysis on Euclidean Spaces.* Princeton University Press, 1971.

[37] F. Oberhettinger, *Fourier Transforms of Distributions and their Inverses: A Collection of Tables.* Academic Press, 1973.