Odyssey 2018 The Speaker and Language Recognition Workshop
26-29 June 2018, Les Sables d'Olonne, France

Supervector Compression Strategies to Speed up I-Vector System Development

Ville Vestman, Tomi Kinnunen

University of Eastern Finland, Finland

vvestman@cs.uef.fi, tkinnu@cs.uef.fi

Abstract

The front-end factor analysis (FEFA), an extension of principal component analysis (PPCA) tailored to be used with Gaussian mixture models (GMMs), is currently the prevalent approach to extract compact utterance-level features (i-vectors) for automatic speaker verification (ASV) systems. Little research has been conducted comparing FEFA to the conventional PPCA applied to maximum a posteriori (MAP) adapted GMM supervectors. We study several alternative methods, including PPCA, factor analysis (FA), and two supervised approaches, supervised PPCA (SPPCA) and the recently proposed probabilistic partial least squares (PPLS), to compress MAP-adapted GMM supervectors. The resulting i-vectors are used in ASV tasks with a probabilistic linear discriminant analysis (PLDA) back-end. We experiment on two different datasets, on the telephone condition of NIST SRE 2010 and on the recent VoxCeleb corpus collected from YouTube videos containing celebrity interviews recorded in various acoustical and technical conditions. The results suggest that, in terms of ASV accuracy, the supervector compression approaches are on a par with FEFA. The supervised approaches did not result in improved performance. In comparison to FEFA, we obtained more than hundred-fold (100x) speedups in the total variability model (TVM) training using the PPCA and FA supervector compression approaches.

1. Introduction

Modern text-independent automatic speaker verification (ASV) relies heavily on the use of *identity vectors* (i-vectors) [1, 2]. I-vectors are compact representations of speech utterances containing useful information for speech-related classification tasks. The i-vector extraction pipeline involves many steps starting from the extraction of acoustic features such as *Mel-frequency cepstral coefficients* (MFCCs), followed by the extraction of *sufficient statistics* with the aid of an *universal background model* (UBM), typically a *Gaussian mixture model* (GMM) [3] or a *deep neural network* (DNN) model [4]. Sufficient statistics are then used to extract an i-vector, a fixed-length representation of an utterance, using a pre-trained *total variability model* (TVM) that models the distribution of utterance-specific GMM supervectors.

The development and optimization of an i-vector based ASV system consists of a multiple time-consuming steps. The most notable ones are extraction of acoustic features and sufficient statistics, and training of UBM and TVM. The two former require processing of a large number of speech files and TVM training is among the most time consuming parts of the system development process. Thus, by reducing TVM training time, a meaningful positive effect to the total development time of ASV system can be achieved [5, 6]. This is particularly beneficial in studies focused on the acoustic front-end optimizations when one has to retrain the entire system when feature extractor is changed [7].

Previous studies on rapid i-vector extraction have primarily optimized computations in the standard *front-end factor analysis* (FEFA) approach [1, 2] by adopting new computational algorithms, often approximative in nature [6, 8, 9]. In this study, however, we focus on an alternative and straightforward compression of classic *maximum a posteriori* (MAP) [3] adapted GMM supervectors with a goal of obtaining fast execution times without compromising on ASV accuracy. In fact, before FEFA, and its predecessor, *joint factor analysis* (JFA) [10], became prevalent, MAP adapted supervectors were commonly used with *support vector machine* (SVM) to do speaker classification [11]. Recently, however, the use of MAP adapted supervectors has been less common.

Supervector compression, for example by using *probabilistic principal component analysis* (PPCA) [12, 13], provides a large computational saving in TVM estimation over FEFA [14]. In FEFA, the posterior covariance matrix needed for i-vector extraction is *utterance-dependent*, while in the supervector compression methods addressed in this study, covariance is shared among all speech utterances, which greatly reduces computation. As the TVM training set may consist of tens of thousands of utterances, the resulting computational saving is considerable [14].

The closest prior work similar in spirit to ours are [14] and [15], which we extend in many ways. In these two studies, the training of TVM is performed using PPCA. The acoustic feature vectors are *hardly aligned* to a single UBM component. If they were *softly aligned*, this approach would equal to using MAP adapted supervectors with a *relevance factor* of 0 [16]. Differing from [14] and [15], we use MAP adapted supervectors to train TVM and we study how the choice of relevance factor affects the system performance.

Recently [17], TVM estimation using *probabilistic partial least squares* (PPLS) was proposed as an alternative to FEFA. PPLS compresses supervectors in a *supervised* way by taking advantage of the speaker labels in the training set. In the current study, we attempt to validate the positive results [17] obtained for Chinese mandarin corpus by using datasets containing English speech instead. In [18], supervision is added to the total variability matrix training by deploying *supervised* PPCA (SPPCA) [19]. The SPPCA model is fundamentally the same as the PPLS model, with a difference in what has been used as the supervision data; PPLS has been used directly with speaker labels [17], while SPPCA has been utilized with speaker-specific (not just utterance-specific) sufficient statistics [18]. In the current work, we study the use of SPPCA for supervector compression.

In addition to the above models, we adopt standard *fac-*

This work was partially funded by Academy of Finland (project 309629).

tor analysis (FA) for supervector compression. Note, that this differs from the FEFA framework: In FEFA, the TVM training is based on the maximization of posterior probabilities of acoustic feature vectors, assumed to have been generated by a GMM [20], whereas in FA (and PPCA), maximization is performed with respect to posterior probabilities of supervectors [21].

We conduct comparisons of different methods in ASV setting using a recently released *VoxCeleb* corpus [22]. The corpus contains "real-world" utterances obtained from YouTube videos of celebrity interviews using a fully automated data collection pipeline. The data is challenging for ASV due to large intra-speaker variability caused by large differences in environments, speaking styles, and technical conditions. In addition to VoxCeleb, we validate our findings with the telephone condition of NIST 2010 Speaker Recognition Evaluation corpus.

Our contributions can be summarized as follows. First, we present all the selected methods in an unified notation highlighting the important formulas regarding their implementation. Second, we aim at validating the results claimed in [17] regarding the recently proposed PPLS method on different corpora, and we extend the study by introducing the weighting scheme proposed in [18]. Third, we implement and test SPPCA in the supervector compression domain. Fourth, we compare all the methods in terms of ASV performances and training times of total variability models. Lastly, we propose a slight simplification to the maximization principle of TVM training.

2. I-Vector Extraction by Front-End Factor Analysis

Front-end factor analysis (FEFA) [1] is the current standard method for extracting utterance level features known as *i-vectors*. In FEFA, a supervector $\boldsymbol{m}(u)$ of an utterance u is modeled as

$$\widehat{\boldsymbol{m}}(u) = \boldsymbol{\mu} + V\boldsymbol{w}(u),$$

where $\boldsymbol{\mu} \in \mathbb{R}^{h\times 1}$ is an utterance-independent bias supervector, $\boldsymbol{w}(u) \in \mathbb{R}^{d\times 1}$ is a low dimensional representation of an utterance supervector, *i.e.*, an i-vector, and $V \in \mathbb{R}^{h\times d}$ is a mapping between low and high dimensional spaces known as *total variability matrix*. The mathematical foundation of FEFA is presented in detail in [20].

The traditional way of TVM estimation and i-vector extraction begins with computing frame posterior probabilities for short-term spectral features (*e.g.* MFCCs) of an utterance with each Gaussian component of UBM. These posteriors are then used to compute zeroth and first order sufficient statistics,

$$n_c = \sum_{t=1}^{T} p_t(c),$$
$$\boldsymbol{f}_c = \sum_{t=1}^{T} p_t(c)\boldsymbol{x_t},$$

where $\boldsymbol{x_t}$ is the t^{th} feature vector of the utterance and $p_t(c)$ is the posterior probability of t^{th} vector belonging to c^{th} component of UBM, computed with the aid of Bayes' rule.

Assuming that the prior distribution $p(\boldsymbol{w}(u))$ is stardard normal, it can be shown [20] that

$$p(\boldsymbol{w}(u)|X(u), V) = \mathcal{N}(\boldsymbol{\mu}(u), \Sigma(u)),$$

where $X(u) = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ is a sequence of all feature vectors in the utterance u and where

I-Vector Extraction

$$\boxed{\begin{aligned} \Sigma(u) &= \left(I + \sum_{c=1}^{C} n_c(u) V_c^{\intercal}\Sigma_c^{-1}V_c\right)^{-1}, \\ \boldsymbol{\mu}(u) &= \Sigma(u)\sum_{c=1}^{C} V_c^{\intercal}\Sigma_c^{-1}(\boldsymbol{f}_c(u) - n_c(u)\boldsymbol{\mu}_c). \end{aligned}}$$

In the above equations, Σ_c is the covariance matrix of the c^{th} UBM component, and V_c and $\boldsymbol{\mu}_c$ are component-wise representations of V and $\boldsymbol{\mu}$ so that

$$V = \begin{bmatrix} V_1 \\ \vdots \\ V_C \end{bmatrix} \quad \text{and} \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_C \end{bmatrix},$$

where the vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C$ are the means of the UBM components. Mean $\boldsymbol{\mu}(u)$ of the posterior i-vector distribution is the i-vector of the utterance u.

The matrix V is estimated using an offline training set of U utterances by maximizing

$$\sum_{u=1}^{U} \mathbb{E}[\ln p(X(u)|\boldsymbol{w}(u), V)], \tag{1}$$

where the expectations are taken with respect to posterior i-vector distributions [20]. This leads to an update formula

Update Formula for V

$$\boxed{\begin{aligned} V_c &= \left(\sum_{u=1}^{U} \boldsymbol{f}_c(u)\boldsymbol{\mu}(u)^{\intercal}\right)\left(\sum_{u=1}^{U} n_c(u)\mathbb{E}_{\mu\mu}(u)\right)^{-1}, \\ \mathbb{E}_{\mu\mu}(u) &= \Sigma(u) + \boldsymbol{\mu}(u)\boldsymbol{\mu}(u)^{\intercal}. \end{aligned}}$$

Training of V is iterative; one iteration consists of computing $\Sigma(u)$, $\boldsymbol{\mu}(u)$, and $\mathbb{E}_{\mu\mu}(u)$ for all training utterances by keeping V fixed and then updating V using the computed values. During the first iteration, parameters of posterior distributions are computed using a randomly initialized matrix V.

3. I-Vector Extraction by Supervector Compression

In this section, we present multiple approaches to compressing MAP adapted GMM supervectors to low-dimensional representations, which we will also refer as "i-vectors". Unlike FEFA, these approaches do not assume the underlying speaker model to be GMM.

In relevance MAP, the adapted mean vectors $\hat{\boldsymbol{\mu}}_c$, $c = 1, \ldots, C$, for utterance's GMM are obtained from UBM by computing

$$\hat{\boldsymbol{\mu}}_c = \alpha_c \boldsymbol{f}_c + (1 - \alpha_c)\boldsymbol{\mu}_c,$$

where

$$\alpha_c = \frac{n_c}{n_c + r}$$

and $r \geq 0$ is the *relevance factor* to be optimized [3]. When $r \to 0$, then $\alpha_c \to 1$, and when $r = 0$, the mean vectors are solely determined by the sufficient statistics $\boldsymbol{f}_c$. If r is large, then the adapted mean vectors remain close to UBM's mean vectors. Adapted mean vectors $\hat{\boldsymbol{\mu}}_c$ are concatenated together to form a *supervector* for the utterance.

3.1. Principal Component Analysis

Being one of the most commonly used dimension reduction techniques, we include the conventional *principal component analysis* (PCA) [23] as a baseline method for supervector compression. The PCA transformation matrix, consisting of eigenvectors of data covariance matrix, can be used to transform high dimensional supervectors to low dimensional i-vectors. In this study, we use the standard *singular value decomposition* (SVD) approach for PCA computation. However, for high dimensional data, PCA could be computed more efficiently by adopting iterative PCA algorithms [12].

3.2. Probabilistic Principal Component Analysis

Probabilistic principal component analysis (PPCA) [12, 13] models observations using a linear-Gaussian framework. In this section, we present a compact self-contained formulation of PPCA in the context of supervectors. The rest of the methods discussed in Section 3 are formulated similarly and their theory can be easily formulated using PPCA as a starting point.

In PPCA, supervectors are modeled as

$$\boldsymbol{m}(u) = V\boldsymbol{w}(u) + \boldsymbol{\epsilon}, \tag{2}$$

where $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, I)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 I)$. For brevity, we have omitted the bias term $\boldsymbol{\mu}$ from the right-hand side of the equation by assuming that supervectors have been centered using the data mean computed from the training data.

By using properties of normally distributed random variables and by assuming that V is given, from (2) it follows that

$$p(\boldsymbol{m}(u)|\boldsymbol{w}(u)) = \mathcal{N}(V\boldsymbol{w}(u), \sigma^2 I).$$

Further, in Appendix A, we show that

$$p(\boldsymbol{w}(u)|\boldsymbol{m}(u)) = \mathcal{N}(\boldsymbol{\mu}(u), \Sigma), \tag{3}$$

where

I-Vector Extraction

$$\boxed{\begin{aligned}\Sigma &= \left(I + \frac{1}{\sigma^2}V^\intercal V\right)^{-1},\\ \boldsymbol{\mu}(u) &= \frac{1}{\sigma^2}\Sigma V^\intercal \boldsymbol{m}(u).\end{aligned}} \tag{4}$$

Unlike with FEFA, covariance Σ of the posterior i-vector distribution does not depend on the utterance. Hence, by adopting PPCA instead of FEFA, the time complexity of computing the parameters of posterior distributions drops from $O(U(CFd + Cd^2 + d^3))$ to $O(UCFd)$, where F is the dimension of acoustic feature vectors [14].

In the current work, we study two different ways of obtaining V for all the presented methods. The traditional approach (**max. principle 1**) maximizes

$$\sum_{u=1}^{U} \mathbb{E}[\ln p(\boldsymbol{m}(u)|\boldsymbol{w}(u), V)], \tag{5}$$

where expectations are taken with respect to posterior i-vector distributions (similar to (1)). We propose maximizing the sum of log-likelihoods directly (**max. principle 2**) without computing expectations by setting $\boldsymbol{w}(u) = \boldsymbol{\mu}(u)$. That is, we maximize

$$\begin{aligned}&\sum_{u=1}^{U} \ln p(\boldsymbol{m}(u)|\boldsymbol{w}(u), V) \qquad (6)\\ &= \sum_{u=1}^{U}\Big(-\frac{h}{2}\ln(2\pi\sigma^2)\\ &\quad -\frac{1}{2\sigma^2}\big(\boldsymbol{m}(u) - V\boldsymbol{\mu}(u)\big)^\intercal\big(\boldsymbol{m}(u) - V\boldsymbol{\mu}(u)\big)\Big)\\ &= -\frac{hU}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{u=1}^{U}\Big(\boldsymbol{m}(u)^\intercal\boldsymbol{m}(u)\\ &\quad - 2\boldsymbol{m}(u)^\intercal V\boldsymbol{\mu}(u) + \boldsymbol{\mu}(u)^\intercal V^\intercal V\boldsymbol{\mu}(u)\Big),\end{aligned}$$

where h is the dimension of supervectors.

By taking derivatives with respect to each variable in the matrix V and by setting them to zero, we obtain

Update Formulas for V and σ^2

$$\boxed{\begin{aligned}V &= \left(\sum_{u=1}^{U}\boldsymbol{m}(u)\boldsymbol{\mu}(u)^\intercal\right)\left(\sum_{u=1}^{U}\mathbb{E}_{\mu\mu}(u)\right)^{-1},\\ \sigma^2 &= \frac{1}{hU}\sum_{u=1}^{U}\Big(\boldsymbol{m}(u)^\intercal\boldsymbol{m}(u) - \operatorname{tr}\big(\mathbb{E}_{\mu\mu}(u)V^\intercal V\big)\Big),\\ \mathbb{E}_{\mu\mu}(u) &= \boldsymbol{\mu}(u)\boldsymbol{\mu}(u)^\intercal \quad \textbf{(max principle 2)}.\end{aligned}}$$

The traditional approach (**max principle 1**) of solving V results in the exact same formulas but with

$$\mathbb{E}_{\mu\mu}(u) = \Sigma + \boldsymbol{\mu}(u)\boldsymbol{\mu}(u)^\intercal \quad [13].$$

Similarly to FEFA, training of V is iterative. As initial values, we use random V and $\sigma^2 = 1$.

3.3. Factor Analysis

Factor analysis (FA) agrees with the model (2) of PPCA, except that the noise term $\boldsymbol{\epsilon}$ has more freedom by letting $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \Psi)$, where $\Psi \in \mathbb{R}^{h\times h}$ is diagonal instead of isotropic [13]. The training procedure is analogous to PPCA [21, pp. 585–586]. First, the parameters of posterior distributions (3) are computed as

I-Vector Extraction

$$\boxed{\begin{aligned}\Sigma &= \left(I + V^\intercal\Psi^{-1}V\right)^{-1},\\ \boldsymbol{\mu}(u) &= \Sigma V^\intercal\Psi^{-1}\boldsymbol{m}(u).\end{aligned}}$$

Then, the model parameters are updated as follows:

Update Formulas for V and Ψ

$$\boxed{\begin{aligned}V &= \left(\sum_{u=1}^{U}\boldsymbol{m}(u)\boldsymbol{\mu}(u)^\intercal\right)\left(\sum_{u=1}^{U}\mathbb{E}_{\mu\mu}(u)\right)^{-1},\\ \Psi &= \frac{1}{U}\sum_{u=1}^{U}\left(\boldsymbol{m}(u)\boldsymbol{m}(u)^\intercal - V\mathbb{E}_{\mu\mu}(u)V^\intercal\right)\odot I,\end{aligned}}$$

where $\odot$ denotes the Hadamard (element-wise) product. The update formula for matrix V is the same as with PPCA.

3.4. Supervised Approaches

The methods presented so far capture the variability between individual utterances regardless of the speaker's identity. That is, their training is unsupervised in the sense that no speaker labels are needed. This makes it convenient to apply any of the above methods to different classification tasks, but leaves an open question whether "better" i-vectors could be extracted by utilizing speaker labels. Thus, we explore two methods that include identity information to the training process of the total variability matrix V to discriminate speakers better. The explored methods, recently proposed *probabilistic partial least squares* (PPLS) [17] and *supervised PPCA* (SPPCA) [18], can both be thought as extensions of the regular PPCA. The underlying models of PPLS and SPPCA are essentially the same, where the difference is in the data used to discriminate speakers: PPLS adds discrimination by using *speaker labels* while SPPCA utilizes *speaker-dependent sufficient statistics* within the FEFA framework. In the current work, however, we apply SPPCA in the supervector context in contrast to [18], where the FEFA context is used.

In PPLS, supervector model is bundled together with a speaker label model. Speaker labels are encoded as *one-hot vectors*, $\boldsymbol{y}(u) \in \mathbb{R}^s$, where s is the number of speakers in the training set. For example, if the utterance u originates from the second speaker of the set, then $\boldsymbol{y}(u) = (0, 1, 0, \dots, 0)^\intercal$. The speaker label model and the supervector model share the same i-vector $\boldsymbol{w}(u)$ as follows:

$$\begin{cases} \boldsymbol{m}(u) = V\boldsymbol{w}(u) + \boldsymbol{\epsilon}, & \text{(supervector model)} \\ \boldsymbol{y}(u) = Q\boldsymbol{w}(u) + \boldsymbol{\zeta}, & \text{(speaker label model)} \end{cases} \tag{7}$$

where (7) is defined in the same way as before, $Q \in \mathbb{R}^{s \times d}$ is a mapping between the i-vector space and the one-hot vector space, and $\boldsymbol{\zeta} \sim \mathcal{N}(\boldsymbol{0}, \rho^2 I)$.

As presented in [17] and [18], the PPLS model leads to

I-Vector Extraction

$$\boxed{\begin{aligned} \Sigma &= \left(I + \frac{1}{\sigma^2}V^\intercal V + \frac{1}{\rho^2}Q^\intercal Q\right)^{-1}, \\ \boldsymbol{\mu}(u) &= \Sigma\left(\frac{1}{\sigma^2}V^\intercal \boldsymbol{m}(u) + \frac{1}{\rho^2}Q^\intercal \boldsymbol{y}(u)\right) \end{aligned}} \tag{8}$$

and

Update Formulas for V, Q, σ^2, and ρ^2

$$\boxed{\begin{aligned} V &= \left(\sum_{u=1}^{U} \boldsymbol{m}(u)\boldsymbol{\mu}(u)^\intercal\right)\left(\sum_{u=1}^{U} \mathbb{E}_{\mu\mu}(u)\right)^{-1}, \\ Q &= \left(\sum_{u=1}^{U} \boldsymbol{y}(u)\boldsymbol{\mu}(u)^\intercal\right)\left(\sum_{u=1}^{U} \mathbb{E}_{\mu\mu}(u)\right)^{-1}, \\ \sigma^2 &= \frac{1}{hU}\sum_{u=1}^{U}\left(\boldsymbol{m}(u)^\intercal \boldsymbol{m}(u) - \operatorname{tr}\left(\mathbb{E}_{\mu\mu}(u)V^\intercal V\right)\right), \\ \rho^2 &= \frac{1}{sU}\sum_{u=1}^{U}\left(\boldsymbol{y}(u)^\intercal \boldsymbol{y}(u) - \operatorname{tr}\left(\mathbb{E}_{\mu\mu}(u)Q^\intercal Q\right)\right), \end{aligned}} \tag{9}$$

where, as before, we assume that all the supervectors and one-hot vectors are centered using the mean vectors calculated from the training data.

To extract i-vectors as in (8), we are required to have speaker information of the utterance stored in $\boldsymbol{y}(u)$. In the testing phase, however, we have no information about the speaker. To this end, we might simply extract the test i-vector using extraction formulas (4) for PPCA [18]. The result is not the same as with PPCA, since the training of V is influenced by the supervised approach. Another solution to deal with the lacking speaker information is to predict speaker labels as a mean of posterior distribution $p(\boldsymbol{y}(u)|\boldsymbol{m}(u))$; see the details in [17]. We found experimentally that both approaches extract exactly the same i-vectors.

The described formulations apply also for SPPCA with a difference that instead of using one-hot encoded speaker labels as vectors $\boldsymbol{y}(s)$, we use speaker dependent supervectors. A speaker dependent supervector is formed by using the acoustic features from all of the speaker's training utterances. Note that with SPPCA, h should be used instead of s in (9) and that Q's dimensionality is the same as V's.

In [18], a *weighted* SPPCA is proposed. In weighted SPPCA, (8) becomes

I-Vector Extraction

$$\boxed{\begin{aligned} \Sigma &= \left(I + \frac{1}{\sigma^2}V^\intercal V + \frac{\beta}{\rho^2}Q^\intercal Q\right)^{-1}, \\ \boldsymbol{\mu}(u) &= \Sigma\left(\frac{1}{\sigma^2}V^\intercal \boldsymbol{m}(u) + \frac{\beta}{\rho^2}Q^\intercal \boldsymbol{y}(u)\right), \end{aligned}}$$

where β is a weight parameter. The weight parameter can be used to adjust the amount of supervision added on top of the conventional PPCA model. With $\beta = 0$, the model equals PPCA and when $\beta = 1$, we have the ordinary SPPCA.

4. Experimental Setup

4.1. VoxCeleb Speaker Recognition Corpus

We performed the speaker verification experiments on the recently published *VoxCeleb* dataset [22]. VoxCeleb contains over 150,000 real-world utterances from 1251 celebrities, of which 561 are females and 690 are males. A key difference to the widely used NIST corpora is that, on average, VoxCeleb has more than 100 utterances per speaker, typically obtained from multiple sessions with highly variable environments and recording conditions providing a large intra-speaker variability. The average utterance length is about 8 seconds. Although most of the utterances are short, the utterance length varies considerably, the longest ones being longer than one minute. Utterances have a sampling rate of 16 kHz.

The dataset was collected using fully automated pipeline that extracts and annotates utterances from YouTube videos. To ensure correct speaker annotation, the pipeline contains automatic face verification verifying that mouth movement in the video corresponds to the audio track and that the speaker's identity is the correct one. The utterances are mostly extracted from interview situations ranging from quiet studios to public speeches in front of large audiences. Differing environments and speaking styles are not the only source of variability, since differences in recording devices and audio processing practices are present in YouTube videos. As the acoustic and technical conditions of the utterances vary considerably, the dataset turns out to be challenging for an automatic speaker verification task as we will see in the experiments.

We used the same standard trial list as in the baseline system of [22]. It contains 40 speakers, whose name starts with the letter 'E'. The list has 37720 trials, half of them (18860) being same speaker trials, which differs substantially from the typical NIST setups with about 10 to 1 ratio between non-target and target trials.

The rest of the speakers were used for developing our speaker verification systems, *i.e*., to train the UBM, TVM and classifier back-end. To speed up experimentation, we utilized only one-fourth of the available training data as we did not see large decrease in system performance by decreasing the amount of training data. Our training set contains total of 37160 utterances from 1211 speakers.

We report recognition accuracy using *equal error rates* (EERs) that are the rates where *false alarm* and *miss* rates are equal. With the current trial list setup, 95% confidence intervals around EERs that are below 8% (the case with most experiments) are at widest $\pm 0.27\%$ absolute. Confidence intervals are computed using *z-test* based methodology presented in [24].

4.2. NIST Data

Even if our primary interest is in the VoxCeleb data, for the sake of reference, we also study different i-vector systems using common condition 5 of NIST 2010 Speaker Recognition Evaluation (SRE10)[1]. Trial segments in condition 5 contain conversational 8 kHz telephone speech spoken with normal vocal effort. The trial list consists of 30373 trials, of which 708 are same speaker trials.

Speaker verification systems were trained using 43308 utterances obtained from SRE04, SRE05, SRE06, Switchboard, and Fisher corpora.

The performances are reported as EERs and *minimum values* of *detection cost function* (minDCF) used in SRE10. The SRE10 detection cost function is given as

$$\text{DCF} = 0.001\, P_{\text{miss}} + 0.999\, P_{\text{fa}},$$

where P_{miss} and P_{fa} are probabilities of miss and false alarm, respectively [25].

4.3. Description of Speaker Verification System

We extracted 38 dimensional acoustic feature vectors containing 19 *Mel-frequency cepstral coefficients* (MFCCs) and their delta coefficients. After discarding features of non-speech segments, we subjected features to utterance-level mean and variance normalization.

The *universal background model* (UBM), 1024 component *Gaussian mixture model* (GMM) with diagonal covariance matrices, was trained using the development data. UBM was used to extract sufficient statistics, which were used in FEFA or in supervector extraction. Supervectors were extracted by first creating utterance specific GMMs using *maximum a posteriori* (MAP) adaptation [3] and then by concatenating mean vectors of adapted GMMs into supervectors.

We extracted 400 dimensional i-vectors, which were then centered, length-normalized, and whitened. Finally, we used simplified *probabilistic linear discriminant analysis* (PLDA) [26] to perform supervised dimensionality reduction of i-vectors into 200 dimensions and to score verification trials.

5. Speaker Verification Experiments

The results presented in Sections 5.1 to 5.6 are given for the VoxCeleb ASV protocol. Section 5.7 presents results for SRE10.

[1]https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2010

5.1. Relevance Factor in MAP Adaptation

We studied how the choice of relevance factor affects speaker verification performance with PPCA and FA methods. The results for VoxCeleb protocol are presented in Figure 1. Based on the results, we fix the relevance factor to $r = 1$ for the remaining experiments with this data. The choice of relevance factor is data-dependent, and therefore, the same value might not work well with other datasets.

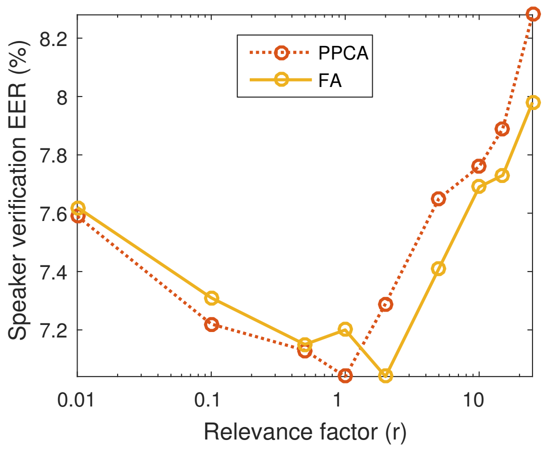

Figure 1: *The effect of relevance factor used in MAP adaptation on speaker verification performance. For VoxCeleb data, relevance factor close to 1 leads to the best results.*

5.2. Number of Training Iterations

To find out the sufficient number of iterations in TVM training, we evaluated verification performances with varying number of iterations. The results of the experiment, presented in Figure 2, reveal that 5 iterations are enough to obtain near to optimal performance. Hence, we fix the number of iterations to 5 for the remaining experiments.

All the methods, except SPPCA, behave similarly. With SPPCA, the training does not proceed in a desirable way during the first 5 iterations.

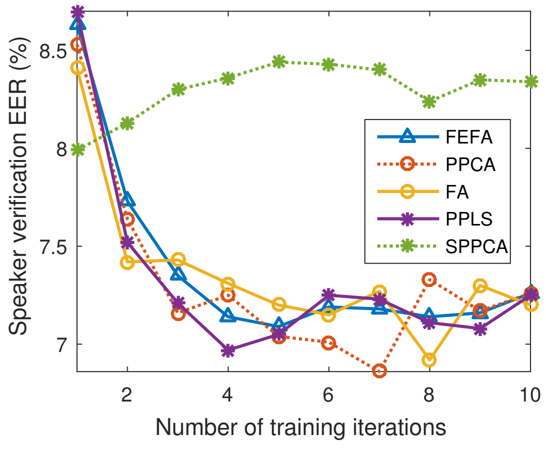

Figure 2: *Speaker verification performances with different numbers of training iterations in total variability model training. About 5 iterations are enough to obtain satisfying performance. The trend is similar for all the methods except for SPPCA, for which the iterative training does not improve the results.*

5.3. Maximization Principles in System Training

In Section 3.2, we presented two different maximization principles that can be used with all the discussed iterative TVM training methods. The comparison of the maximization principles in terms of resulting speaker verification performances is presented in Table 1. There are no clear differences between the principles.

Table 1: *Speaker verification equal error rates (%) for different methods and maximization principles used in TVM training. With conventional PCA approach we obtained EER of 7.39%.*

	max1 [Eq. (5)]	max2 [Eq. (6)]
FEFA	7.09	7.11
PPCA	**7.04**	7.18
FA	7.20	7.26
PPLS	7.05	7.42
SPPCA	8.44	8.26

5.4. Training Times

Figure 3 shows the elapsed times for 5 TVM training iterations with different methods. The measured times do not include the times needed to compute sufficient statistics or supervectors or to load them into memory using I/O operations. Measurements were conducted by running MATLAB implementations of all the methods in a 16-core 2.60 GHz machine with an ample amount of RAM (>300GB). To obtain reasonable training time with FEFA, we trained the system with 8 CPU cores, whereas other methods were trained using a single core.

Before the TVM training phase, different methods have only small differences in terms of computational requirements. Even though FEFA differs from other methods in that it uses sufficient statistics as inputs, the difference is minuscule, as most of the time in the extraction of MAP adapted supervectors is spent in the computation of sufficient statistics.

From the perspective of system optimization, note that FEFA does not require optimization of the relevance factor. But, the extra cost of relevance factor optimization in PPCA-PLDA system does not outweigh the training time of FEFA, as MAP adaptation using precomputed sufficient statistics and training of PPCA and PLDA are much less expensive operations than FEFA training.

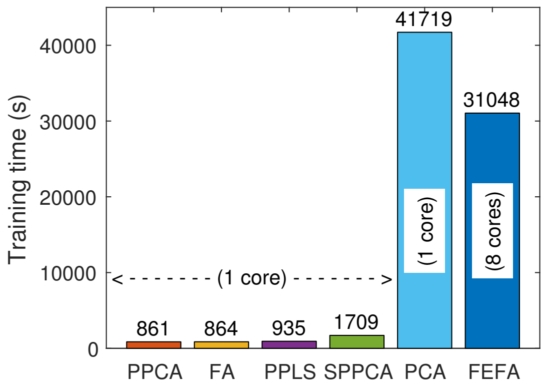

Figure 3: *Training times of TVMs with 5 iterations (PCA is non-iterative). Iterative supervector compression methods are the fastest to train, while FEFA requires the most amount of computation. FEFA was trained with 8 CPU cores to reduce the training time.*

5.5. Weight Parameter in Supervised Approaches

Next, we apply weighting to the supervised methods, PPLS and SPPCA, as discussed in Section 3.4. The results obtained with different weight parameter values are presented in Figure 4. We find that weighting does not affect PPLS and that the weighted SPPCA model functions better when it approaches PPCA ($\beta \to 0$). This suggests that the studied supervised methods do not provide any noticeable benefits over the unsupervised i-vector extractors.

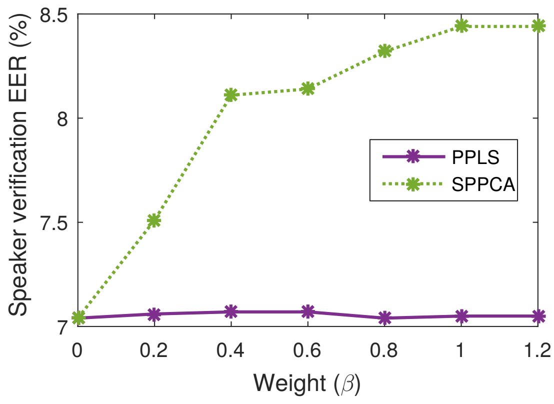

Figure 4: *Effect of the weight parameter (β) value in the supervised models. When $\beta = 0$, both models equal to the conventional PPCA. Adding supervision by increasing β does not improve the speaker verification performance.*

5.6. Dimensionality of I-Vectors

To improve the speaker verification performance, we jointly optimized dimensions of i-vectors and their PLDA-compressed versions. We varied the i-vector dimensionality between 200 and 1000 and the PLDA subspace dimensionality between 100 and 500. The results for all combinations using PPCA method are presented in Figure 5. The results indicate that, our initial parameters, 400 and 200 for i-vectors and PLDA, respectively, give a relatively good performance. We also see that a slight increase in performance might be obtained by using i-vector dimensions between 600 and 1000 with 350 to 400 dimensional PLDA subspaces.

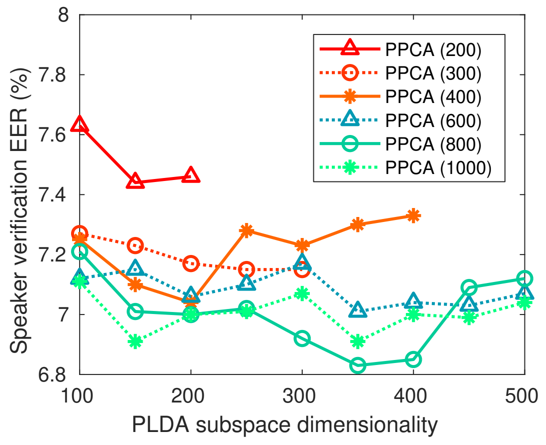

Figure 5: *Optimization of i-vector and PLDA subspace dimensions for PPCA method. Different lines represent different i-vector dimensionalities. The lowest error rate is obtained with 800-dimensional i-vectors and 350-dimensional PLDA space.*

5.7. Evaluation With NIST SRE10 Data

To increase our confidence of the generality of the results, we tested different i-vector systems on common condition 5 of SRE10. We ran the second protocol using mostly the same system configuration as with VoxCeleb corpus. We changed the filterbank spacing in MFCCs to match the 8 kHz sampling rate and we also increased the relevance factor to 6. Otherwise, the system was not optimized for the new data.

The results for SRE10, shown in Table 2, indicate that there are no clear differences between the two maximization principles. Further, we observe that the results for EER and minDCF are somewhat different as FEFA performs the best in terms of EER, while FA obtains the lowest minDCF. To gain better understanding on the performances of the systems, in Figure 6, we present *detection error trade-off* (DET) curves for all the methods using the maximization principle 1. The DET curves reveal that there are no clear differences between FEFA and FA.

6. Discussion and Conclusions

The development and optimization of i-vector systems tends to be time consuming. In particular, any change in the acoustic front-end or the UBM configuration requires retraining the TVM. If TVM training is slow, the parameter optimization can become unfeasible, possibly leading to suboptimal system configuration. In this work, we studied fast GMM supervector compression methods to streamline ASV system development. By focusing on compression of MAP adapted supervectors, we managed to cut the system training time down to a fraction of traditional approach (FEFA). Our results indicate that the alternative approaches work as well as the standard FEFA in terms of recognition accuracy. The less-optimistic performance reported in [14] and [15] (for the PPCA system) could be due to absence of MAP adaptation: we found that increasing the relevance factor from 0 (no MAP adaptation) towards some higher (optimized) value results in considerably higher verification accuracy.

We did not find benefit with either of the studied supervised models, PPLS or SPPCA. We were not, therefore, able to reproduce positive findings claimed in [17] for PPLS. This might be due to differing datasets or feature configurations. On a positive side, we found that PPLS attains similar training speeds to PPCA and FA.

The proposed modification to the maximization principle in TVM training did not affect verification results negatively. This modification makes the theory and the system implementation slightly simpler as it avoids computing expectations over i-vector posterior distributions.

We recognize that the findings of the current study can not be generalized to all existing system configurations without further studies. In this study, we only experimented with a specific set of acoustic features together with a specific UBM and backend configurations (PLDA).

In summary, from the various compared variants, we recommend to use PPCA and FA to compress supervectors. Both are easy to implement on top of the GMM framework and lead to considerably faster TVM training times. For optimal verification accuracy, supervectors should be created using the MAP adaptation with an optimized relevance factor. We have made our MATLAB implementations of PPCA, FA, PPLS, and SPPCA available at `http://cs.uef.fi/~vvestman/research_codes/supervector_compression.zip`.

Table 2: *Speaker verification performances for different methods and maximization principles (max1, max2) on common condition 5 of SRE10. With conventional PCA we obtained EER of 4.69% and minDCF of 6.55%.*

	EER (%)		minDCF (%)	
	max1	max2	max1	max2
FEFA	4.24	**3.86**	6.24	6.85
PPCA	4.66	4.66	6.55	6.72
FA	4.24	4.24	6.00	**5.50**
PPLS	4.66	4.79	6.73	6.53
SPPCA	4.46	4.38	6.46	6.26

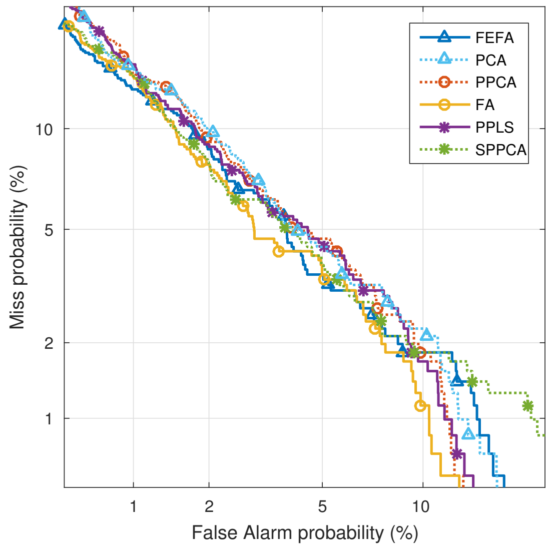

Figure 6: *Detection error trade-off curves for different methods using max. principle 1 on common condition 5 of SRE10.*

7. References

[1] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] Patrick Kenny, "A small footprint i-vector extractor," in *Odyssey*, 2012, vol. 2012, pp. 1–6.

[3] Douglas Reynolds, Thomas Quatieri, and Robert Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[4] Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *ICASSP 2014*. IEEE, pp. 1695–1699.

[5] Ondřej Glembek, Lukáš Burget, Pavel Matějka, Martin Karafiát, and Patrick Kenny, "Simplification and optimization of i-vector extraction," in *ICASSP 2011*, pp. 4516–4519.

[6] Sandro Cumani and Pietro Laface, "Memory and computation trade-offs for efficient i-vector extraction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 934–944, 2013.

[7] Ville Vestman, Dhananjaya Gowda, Md Sahidullah, Paavo Alku, and Tomi Kinnunen, "Time-varying autoregressions for speaker verification in reverberant conditions," in *Interspeech 2017*.

[8] Sandro Cumani and Pietro Laface, "Factorized sub-space estimation for fast and memory effective i-vector extraction," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 1, pp. 248–259, 2014.

[9] Longting Xu, Kong Aik Lee, Haizhou Li, and Zhen Yang, "Generalizing i-vector estimation for rapid speaker recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2018.

[10] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[11] William Campbell, Douglas Sturim, and Douglas Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE signal processing letters*, vol. 13, no. 5, pp. 308–311, 2006.

[12] Sam Roweis, "EM algorithms for PCA and SPCA," in *Advances in neural information processing systems*, 1998, pp. 626–632.

[13] Michael Tipping and Christopher Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[14] Srikanth Madikeri, "A fast and scalable hybrid FA/PPCA-based framework for speaker recognition," *Digital Signal Processing*, vol. 32, pp. 137–145, 2014.

[15] Srikanth Madikeri, "A hybrid factor analysis and probabilistic pca-based system for dictionary learning and encoding for robust speaker recognition," in *Odyssey 2012*, pp. 14–20.

[16] Bavattichalil Haris and Rohit Sinha, "On exploring the similarity and fusion of i-vector and sparse representation based speaker verification systems," in *Odyssey 2012*, pp. 21–27.

[17] Chen Chen, Jiqing Han, and Yilin Pan, "Speaker verification via estimating total variability space using probabilistic partial least squares," in *Interspeech 2017*, pp. 1537–1541.

[18] Yun Lei and John Hansen, "Speaker recognition using supervised probabilistic principal component analysis," in *Interspeech 2010*, pp. 382–385.

[19] Shipeng Yu, Kai Yu, Volker Tresp, Hans-Peter Kriegel, and Mingrui Wu, "Supervised probabilistic principal component analysis," in *KDD '06*. ACM, 2006, pp. 464–473.

[20] Patrick Kenny, Gilles Boulianne, and Pierre Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE transactions on speech and audio processing*, vol. 13, no. 3, pp. 345–354, 2005.

[21] Christopher Bishop, "Machine learning and pattern recognition," *Springer*, 2006.

[22] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, "VoxCeleb: A large-scale speaker identification dataset," in *Interspeech 2017*, pp. 2616–2620.

[23] Karl Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[24] Samy Bengio and Johnny Mariéthoz, "A statistical significance test for person authentication," in *Odyssey 2004: The Speaker and Language Recognition Workshop*, pp. 237–244.

[25] Alvin Martin and Craig Greenberg, "The NIST 2010 Speaker Recognition Evaluation," in *Interspeech 2010*, pp. 2726–2729.

[26] Daniel Garcia-Romero and Carol Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Interspeech 2011*.

A. Proof: I-Vector Posterior Distribution for PPCA Model

The following proof is similar in principle to the proof of Proposition 1 in [20].

It is enough to show that $p(\boldsymbol{w}(u)|\boldsymbol{m}(u)) \propto \mathcal{N}(\boldsymbol{\mu}(u), \Sigma)$, since $p(\boldsymbol{w}(u)|\boldsymbol{m}(u))$ is a probability density function and will hence be correctly scaled. For brevity, we drop u from the notation of the following chain of relations that proves the claim (*e.g.* $\boldsymbol{\mu}$ will refer to $\boldsymbol{\mu}(u)$):

$$\begin{aligned}
&\mathcal{N}(\boldsymbol{\mu}, \Sigma) \\
&\propto \exp\Big(-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^\intercal \Sigma^{-1}(\boldsymbol{w}-\boldsymbol{\mu})\Big) \\
&= \exp\Big(-\frac{1}{2}\Big(\boldsymbol{w}-\frac{1}{\sigma^2}\Sigma V^\intercal \boldsymbol{m}\Big)^\intercal \Big(I+\frac{1}{\sigma^2}V^\intercal V\Big) \\
&\quad \Big(\boldsymbol{w}-\frac{1}{\sigma^2}\Sigma V^\intercal \boldsymbol{m}\Big)\Big) \\
&= \exp\Big(-\frac{1}{2}\Big[\Big(\boldsymbol{w}^\intercal+\frac{1}{\sigma^2}\boldsymbol{w}^\intercal V^\intercal V-\frac{1}{\sigma^2}(\Sigma V^\intercal \boldsymbol{m})^\intercal \Sigma^{-1}\Big) \\
&\quad \Big(\boldsymbol{w}-\frac{1}{\sigma^2}\Sigma V^\intercal \boldsymbol{m}\Big)\Big]\Big) \\
&= \exp\Big(-\frac{1}{2}\Big[\boldsymbol{w}^\intercal \boldsymbol{w}+\frac{1}{\sigma^2}\boldsymbol{w}^\intercal V^\intercal V\boldsymbol{w} \\
&\quad -\frac{1}{\sigma^2}\Big(\Sigma V^\intercal \boldsymbol{m}\Big)^\intercal \Sigma^{-1}\boldsymbol{w} \\
&\quad -\Big(\boldsymbol{w}^\intercal+\frac{1}{\sigma^2}\boldsymbol{w}^\intercal V^\intercal V\Big)\frac{1}{\sigma^2}\Sigma V^\intercal \boldsymbol{m}\Big]+\mathrm{const}(\boldsymbol{m})\Big) \\
&\propto \mathcal{N}(\boldsymbol{0}, I)\exp\Big(-\frac{1}{2\sigma^2}\Big[(V\boldsymbol{w})^\intercal V\boldsymbol{w}-\boldsymbol{m}^\intercal V\Sigma^\intercal \Sigma^{-1}\boldsymbol{w} \\
&\quad -\boldsymbol{w}^\intercal \Sigma^{-1}\Sigma V^\intercal \boldsymbol{m}\Big]\Big) \\
&\propto \mathcal{N}(\boldsymbol{0}, I)\exp\Big(-\frac{1}{2\sigma^2}\Big[(V\boldsymbol{w})^\intercal V\boldsymbol{w}-\boldsymbol{m}^\intercal V\boldsymbol{w} \\
&\quad -(V\boldsymbol{w})^\intercal \boldsymbol{m}+\boldsymbol{m}^\intercal \boldsymbol{m}\Big]\Big) \\
&= \mathcal{N}(\boldsymbol{0}, I)\exp\Big(-\frac{1}{2}(\boldsymbol{m}-V\boldsymbol{w})^\intercal \frac{1}{\sigma^2}(\boldsymbol{m}-V\boldsymbol{w})\Big) \\
&\propto \frac{p(\boldsymbol{w})p(\boldsymbol{m}|\boldsymbol{w})}{p(\boldsymbol{m})} \\
&= p(\boldsymbol{w}|\boldsymbol{m}).
\end{aligned}$$

Note that in the above chain of proportional relations, we can drop $(\exp(\mathrm{const}(\boldsymbol{m})))$ and add $(\exp(\boldsymbol{m}^\intercal \boldsymbol{m})$; $p(\boldsymbol{m}))$ multipliers that only depend on $\boldsymbol{m}$ without breaking the chain.