In the testing phase, given a set of testing vectors $\{x_1, x_2, \ldots, x_T\}$, UBM model S^{ubm} and the adapt models of each language $\{S_i^{adapt}\}_{i=1}^{M}$, the identified language model can be evaluated using equation 2.34. Where M is the total no of languages used in the system.

$$\hat{S} = \arg \max_{1 \leq i \leq M} \sum_{j=1}^{T} \left[\log(P(x_j|S_i^{adapt})) - \log(P(x_j|S^{ubm}) \right] \tag{2.34}$$

2.6 I-vector based language identification system

Language recognition and speaker recognition has many similarities in terms of technical formulation. Thus, in literature most of the descriptors and modeling techniques used to perform language identification task are borrowed from speaker recognition. In case of speaker recognition, in GMM-UBM based statistical modeling, the MAP adaptation not only adapt the speaker specific information but also adapts the channel and session variations. Thus to improve the speaker recognition performance, needs a modeling technique which can model the speaker specific information and other variability separately or to model all the information, suppress the other variability and retain the speaker specific information. In [48], the authors propose a method to model all the variations separately. They represented a speaker utterance by a supervector (M), that consists of additive components of speaker and channel/session subspace. This technique is well known as joint factor analysis (JFA). The speaker dependent supervector is defined as:

$$M = m + Vy + Ux + Dz \tag{2.35}$$

where m is a speaker and session independent supervector (generally mean vector of UBM having $M_g D \times 1$ dimension). V and D define a speaker subspace (eigenvoice matrix and diagonal residual respectively), and U defines a session or channel subspace (eigen channel matrix). The vector y, z and x are the speaker and channel/session dependent factors in their respective subspace and each is assumed to be a random variable with a standard normal distribution ($N(0, I)$). In JFA, first the subspaces (i.e V, D and U) have to be estimated from the appropriate labeled corpora, than the speaker and session factors (i.e y, z and x) have to be computed for the target utterances. The scoring is done by computing the likelihood of the test utterance vectors against the session compensated speaker model ($M - Ux$). In [49], the authors slightly modify the modeling technique to reduce the computation and enhance the recognition performance. The authors proposed a single subspace modeling instead of two subspace (speaker and channel separately). This single subspace is known as total variability subspace. The total variability subspace contains both the speaker and channel variability information. The new speaker and

channel dependent supervector can be defined as:

$$\mathrm{M} = \mathrm{m} + \mathrm{Tw} \tag{2.36}$$

where T is the total variability subspace and w is the speaker and channel dependent vector (known as identity vector/ i-vector). T is a rectangular matrix of low rank and w is a vector having standard normal distribution. After i-vector extraction some channel compensation techniques (like linear discriminative analysis (LDA), within class covariance normalization (WCCN) and nuisance attribute projection (NCA)) are used to suppress the channel variability. A cosine kernel scoring technique is used to find the recognition performance. The block diagram of i-vector based language recognition system is shown in figure 2.9.

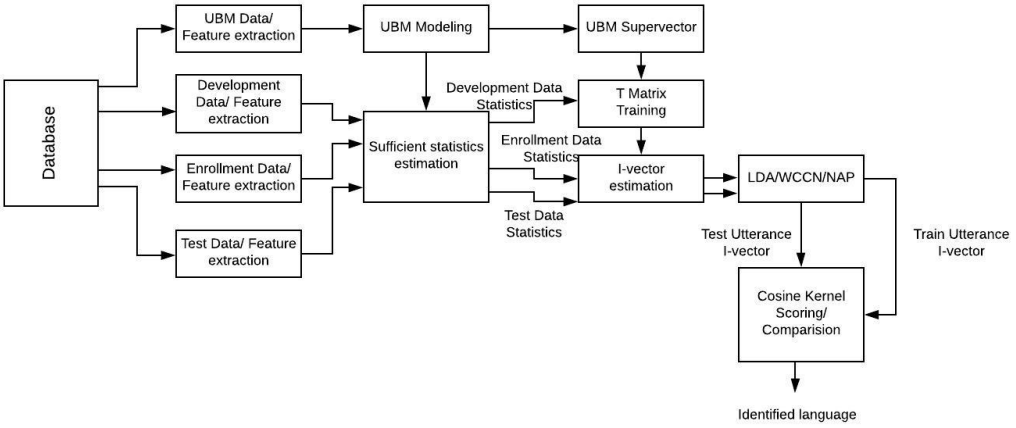

FIGURE 2.9: Basic block diagram of language identification using i-vector.

From the figure 2.9, it has been seen that, the whole dataset is segmented into four parts: UBM data, development data, enrollment data, test data. UBM data consists of data of all the languages, used to build UBM models as described in section 2.5. The development data segment also consists of utterances from all the languages, used to estimate the total variability matrix. After the estimation of total variability matrix (T matrix), the hidden variable w, known as i-vector is estimated from each utterance of the enrollment and test data. The i-vector w, which can be defined by its posterior distribution conditioned to the Baum-Welch statistics for a given utterance. The posterior distribution is a Gaussian distribution and the Baum-Welch sufficient statistics will be estimated from the UBM. Suppose we have T frames $\{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{T}\}$ of an utterance and the UBM composed of M_g mixture components defined in some feature space of

dimension D. The Sufficient statistics for a given utterance is given by:

$$\begin{aligned} N_i &= \sum_{j=1}^{T} \gamma_{ij} \quad \text{(zeroth order statistics)} \\ F_i &= \sum_{j=1}^{T} \gamma_{ij} x_j \quad \text{(first order statistics)} \\ S_i &= \text{diag}\left[\sum_{j=1}^{T} \gamma_{ij} x_j x_j^*\right], \quad i = 1, 2, \ldots, M_g \quad \text{(second order statistics)} \end{aligned} \tag{2.37}$$

where x_j^* is the Hermitian transpose of vector x_j and $\gamma_{ij} = \frac{w_i P(x_j|\theta_i)}{\sum_{i=1}^{M_g} w_i P(x_j|\theta_i)}$ (from equation 2.21). In order to estimate the i-vector, we need to centralize the first and second order sufficient statistics. The centralized statistics is given by:

$$\begin{aligned} \tilde{F}_i &= \sum_{j=1}^{T} \gamma_{ij}(x_j - m_i) \\ &= \sum_{j=1}^{T} \gamma_{ij} x_j - \sum_{j=1}^{T} \gamma_{ij} m_i \\ &= F_i - N_i m_i \\ \tilde{S}_i &= \text{diag}\left[\sum_{j=1}^{T} \gamma_{ij}(x_j - m_i)(x_j - m_i)^*\right], \quad i = 1, 2, \ldots, M_g \\ &= S_i - \text{diag}(F_i m_i^* + m_i F_i^* - N_i m_i m_i^*) \end{aligned} \tag{2.38}$$

where m_i is the mean vector of the i^{th} mixture. We can write the statistics in the matrix format as:

$$\begin{aligned} N_m &= \begin{bmatrix} N_1 * I & & \\ & \ddots & \\ & & N_{M_g} * I \end{bmatrix} \\ F_m &= \begin{bmatrix} \tilde{F}_1 \\ \vdots \\ \tilde{F}_{M_g} \end{bmatrix} \\ S_m &= \begin{bmatrix} \tilde{S}_1 & & \\ & \ddots & \\ & & \tilde{S}_{M_g} \end{bmatrix} \end{aligned} \tag{2.39}$$

where N_m is a matrix of dimension $M_g D \times M_g D$, I is a identity matrix of dimension $D \times D$, F_m is a vector of dimension $M_g D \times 1$ and S_m is a matrix of dimension $M_g D \times M_g D$. In [48] from Theorem 2 we can write the distribution of the hidden variable (w) is $N(L_T^{-1} * T^* * \Sigma^{-1} * F_m, L_T^{-1})$. The i-vector can be computed as $E(w) = L_T^{-1} * T^* * \Sigma^{-1} * F_m$. where E(w) is the expectation

of vector w, Σ is the variance of the UBM, $L_T = I + T^*\Sigma^{-1}N_mT$ and T is the total variability matrix.

2.6.1 T Matrix training

1. Step 1: Initialize the T matrix randomly having dimension $M_gD \times$ Dimension of $i-$vector
2. Step 2: Compute: $L_T = I + T^*\Sigma^{-1} * N_m * T$
3. Step 3: Accumulate the statistics across utterances:

$$\begin{aligned} {}^aN_i &= \sum_u N_i(u) \\ {}^aA_i &= \sum_u N_i(u)L_T^{-1} \\ {}^aC &= \sum_u F_m(u)(L_T^{-1}(u) * T^* * \Sigma^{-1} * F_m(u)) \\ {}^aN_m &= \sum_u N_m(u) \end{aligned} \tag{2.40}$$

4. Step 4: Compute T matrix:

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_{M_g} \end{bmatrix} = \begin{bmatrix} A_1^{-1} * C_1 \\ \vdots \\ A_{M_g^{-1}} * C_{M_g} \end{bmatrix}, \quad \text{where} \quad C = \begin{bmatrix} C_1 \\ \vdots \\ C_{M_g} \end{bmatrix} \tag{2.41}$$

5. Step 5: Compute covariance update (optional):

$$\Sigma = ({}^aN_m)^{-1}\left[\sum_u S_m(u) - \text{diag}(C * T^*)\right] \tag{2.42}$$

6. Step 6: Iterate step 2 to step 4 (or step 5) approximately 20 times to get the estimate of the T matrix.

In the literature, people have not used the second order statistics(S_m) for speaker and language recognition task, in such cases step 5 is not required.

2.6.2 I-vector estimation

After the estimation of T matrix the i-vectors from each utterances of the enrollment set and test set is computed using equation 2.43.

$$w(u) = L_T^{-1}(u) * T^* * \Sigma^{-1} * F_m(u) \tag{2.43}$$

2.6.3 Cosine kernel scoring

Cosine kernel scoring is technique to find the cosine kernel between the train language i-vectors and test i-vector. Suppose there are L languages having each $K_1, K_2, \ldots, K_L$ utterances and lets denote the train language i-vector as w_{lk} and the test utterance i-vector as w_{test}. The identified language ($\hat{S}$) can be written as

$$\hat{S} = \arg \max_{1 \leq l \leq L} \sum_{k=1}^{K_l} Score_{lk} = \arg \max_{1 \leq l \leq L} \sum_{k=1}^{K_l} \frac{< w_{lk}, w_{test} >}{||w_{lk}||||w_{test}||} \tag{2.44}$$

2.6.4 Intersession compensation techniques

The intersession compensation techniques are used to enhance the inter-class variations and to suppress the intra-class variations. Generally there are three dominantly used intersession compensation technique to enhance the recognition performance, while the classes are modeled in total variability space. These three techniques are: linear discriminant analysis (LDA), within class covariance normalization (WCCN), nuisance attribute projection (NAP).

2.6.4.1 Linear discriminant analysis (LDA)

LDA is a technique of dimensionality reduction, widely used in the task of pattern recognition. The motivation of using LDA in i-vector based language recognition task is that, as all the utterances of a given language are assume to represent one class, LDA attempts to define new special axes that minimize the intra-class variance caused by channel and speaker effects, and to maximize the variance between languages. This approach searches a new orthogonal axes to better discriminate between classes. The LDA optimization problem can be defined according to the ratio given in equation 2.45.

$$J(v) = \frac{v^t S_b v}{v^t S_w v} \tag{2.45}$$

The ratio is often referred as the Rayleigh coefficient for space direction v. S_b and S_w are the between class and the within-class variance between matrix and are calculated as follows:

$$\begin{aligned} S_b &= \sum_{l=1}^{L} (w_l - \bar{w})(w_l - \bar{w})^t \\ S_w &= \sum_{l=1}^{L} \frac{1}{K_L} \sum_{k=1}^{K_L} (w_k^l - \bar{w}_l)(w_k^l - \bar{w}_l)^t \end{aligned} \tag{2.46}$$

where $\bar{w}_l = \frac{1}{K_L} \sum_{k=1}^{K_L} w_k^l$ is the mean of i-vectors for each language. $\bar{w}$ is the mean of i-vector for all the utterances. The maximization is used to define a projection matrix A composed by

the best eigen vectors (those with highest eigen values) of the general eigen value equation:

$$\begin{aligned} S_b v &= \lambda S_w v \\ S_b S_w^{-1} v &= \lambda v \end{aligned} \tag{2.47}$$

where λ is the diagonal matrix of eigenvalues. The the projection matrix is obtained by taking the eigen vectors of the larger eigenvalues (i.e $A = V(:, 1 : \text{no of top eigen values})$).The identified language ($\hat{S}$) can be written as:

$$\hat{S} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} Score_{lk} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \frac{< A^t w_{lk}, A^t w_{test} >}{||A^t w_{lk}||||A^t w_{test}||} \tag{2.48}$$

2.6.4.2 Within class covariance normalization (WCCN)

This technique was first introduced in [50]. In the study, they applied this approach in SVM modeling based on linear separation between target speaker and imposter using one verses all decision. WCCN techniques uses inverse of within-class covariance to normalize the cosine kernel. The resulting solution by a generalized linear kernel form can be written as given in [50]:

$$k(w_1, w_2) = w_1^t R w_2 \tag{2.49}$$

where R is a symmetric positive semi-definite matrix. The optimal normalized kernel matrix is given by $R = W^{-1}$. W is a within class covariance matrix, calculated as:

$$W = \frac{1}{L} \sum_{l=1}^{L} \frac{1}{K_L} \sum_{k=1}^{K_L} (w_k^l - \bar{w}_l)(w_k^l - \bar{w}_l)^t \tag{2.50}$$

In order to preserve the inner product form of the cosine kernel, a feature-mapping function φ can be defined as:

$$\varphi(w) = B^t w \tag{2.51}$$

where B is obtained through Cholesky decomposition of matrix $W^{-1} = BB^t$. The WCCN algorithm uses the within-class covariance matrix to normalize the cosine kernel function to normalize the cosine kernel functions in order to compensate for intersession variability, while guaranteeing conservation of directions in space. The identified language ($\hat{S}$) can be written as:

$$\hat{S} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} Score_{lk} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \frac{< B^t w_{lk}, B^t w_{test} >}{||B^t w_{lk}||||B^t w_{test}||} \tag{2.52}$$

2.6.4.3 Nuisance attribute projection (NAP)

The nuisance attribute projection algorithm is presented in [51]. It is based on finding an appropriate projection matrix intended to remove the nuisance direction. The projection matrix carries out an orthogonal projection in channel's and speaker's complimentary space, which depends only on language class information. The projection matrix is formulated as:

$$\mathrm{P} = \mathrm{I} - \mathrm{R}\mathrm{R}^{\mathrm{t}} \tag{2.53}$$

where R is a rectangular matrix of low rank, whose columns are the eigen vectors ($\mathrm{Wr} = \lambda \mathrm{r}$) having the best eigen values of the within-class covariance matrix (W) computed as:

$$\mathrm{W} = \frac{1}{\mathrm{L}} \sum_{\mathrm{l}=1}^{\mathrm{L}} \frac{1}{\mathrm{K_L}} \sum_{\mathrm{k}=1}^{\mathrm{K_L}} (\mathrm{w}_{\mathrm{k}}^{\mathrm{l}} - \bar{\mathrm{w}}_{\mathrm{l}})(\mathrm{w}_{\mathrm{k}}^{\mathrm{l}} - \bar{\mathrm{w}}_{\mathrm{l}})^{\mathrm{t}} \tag{2.54}$$

The identified language ($\hat{\mathrm{S}}$) using NAP can be written as:

$$\hat{\mathrm{S}} = \arg \max_{1 \le \mathrm{l} \le \mathrm{L}} \sum_{\mathrm{k}=1}^{\mathrm{K_l}} \mathrm{Score}_{\mathrm{lk}} = \arg \max_{1 \le \mathrm{l} \le \mathrm{L}} \sum_{\mathrm{k}=1}^{\mathrm{K_l}} \frac{< \mathrm{P}^{\mathrm{t}} \mathrm{w}_{\mathrm{lk}}, \mathrm{P}^{\mathrm{t}} \mathrm{w}_{\mathrm{test}} >}{||\mathrm{P}^{\mathrm{t}} \mathrm{w}_{\mathrm{lk}}|| ||\mathrm{P}^{\mathrm{t}} \mathrm{w}_{\mathrm{test}}||} \tag{2.55}$$

2.7 Feedforward network based model

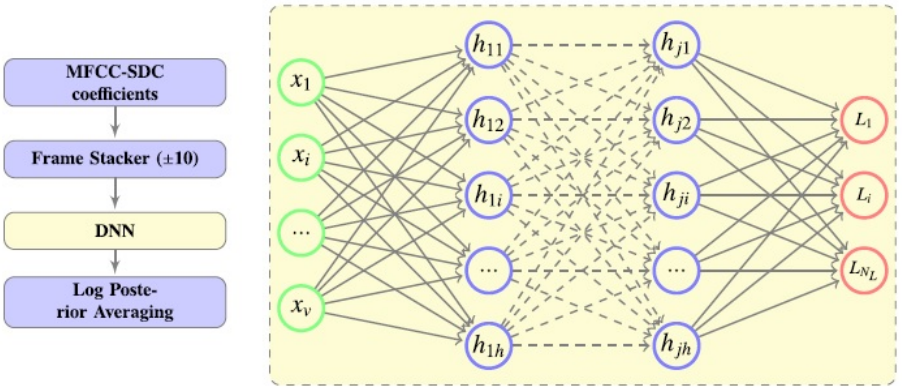

FIGURE 2.10: Feedforward network based classifier [3].

The impressive gain in performance obtained using deep neural networks (DNNs) for automatic speech recognition (ASR) have motivated the application of DNNs in other speech technologies such as language recognition and speaker recognition. Feedforward neural network is a fully connected network, which shows the direct use of DNN in the recognition task. In this approach, a fully connected network with multiple hidden layers is trained using back-propagation algorithm to perform recognition task. The output layer of this network has the number of neurons (nodes)