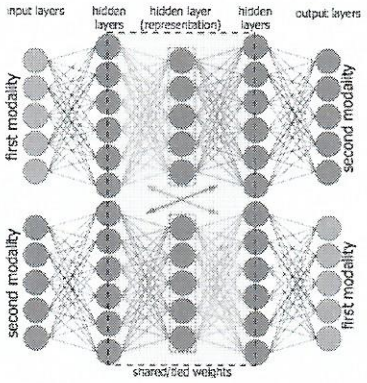
Figure 5. Symmetrical and bidirectional learning architecture using tied weights




## IV. Method

As explained earlier, supervectors computed from Gaussian Mixture Models provide useful features of a given signal. i-vectors have built upon this representation to extract a more compact reprensentation that better reprensent the specificities of the signal.

One task that naturally comes to mind is deciding whether two signals have been said by the same person. Unfortunately, computing euclidian distance between the signals' respective supervector or i-vector does not yield satisfactory results. Therefore, we will use neural networks to extract an intermediate representation of each signal that might be better suited to this particular problem.

### A. Prospective methodology

*a) Evaluating speaker-dependant similarities:* We wish to acquire intermediate representations that draws different signals spoken by the same locutor closer together according to some measure of distance, therefore allowing for an easy classification. The distance we will mostly consider during our study will be angular distance and euclidian distance.

To this end, we will have to carefully chose the distance function used to evaluate the "quality" of our output in order to make sure we will be optimizing this particular aspect of the acquisition.

*b) Phasing out non-speaker dependant noise:* In order to extract a representation of supervectors that makes speaker dependent features of the given signal more salient, we use a model derived from the design of a denoising autoencoder. Indeed, we have no need for any information pertaining to what is actually being said or what kind of microphone was used: this is all noise to us. Therefore, if two supervectors originate from the same speaker, it seems reasonnable to think of them as the same original "speaker" sound vector which has polluted by non-speaker dependant noise.

The latent representation thusly extracted should present salient features specific to the studied speaker. An autoencoder trained that way could allow for a sort of projection that we wish to study.

*c) Prospective general model:* We will therefore train an autoencoder on supervectors. Given good results of [11], we will tie all but the extreme weights to reflect the symmetrical nature of the task at hand.

### B. First experimentation

In light of the difficulties inherent to the training of large neural networks, a preliminary study on a small dataset will first be conducted.

*1) Dataset:*

*a) Processing raw data:* 15311 audio signals were extracted from BFMTV's various programs and labeled with the name of their respective speaker. Those soundfiles were then processed into 15311 supervectors of length 9216 with the *AudioSeg* tool. We then create 1 839 235 pairs of supervectors that originate from the same speaker.

*b) Input:* For each computed pair, we feed both of its supervectors to the neural network as input. Therefore, we train the network on 3 678 470 supervectors of size 9216.

*c) Output and task:* The neural network outputs a vector of size 9216 that we try to match as closely as possible to the supervector the original input was paired with.

*2) Neural approach:* We will start by training a 7-hidden layer auto-encoder.

The encoder is 3 layer deep and encodes the input of size 9216 into a latent representation of size 40. This is done by successively compressing into representations of size 5000 (first layer), 500 (second layer) and 100 (third layer).

The decoder is 3 layer deep and inflates the 40 dimensional latent representation back into a 9216 dimensional supervector by successively inflating into representations of size 100 (fifth layer), 500 (sixth layer) and 5000 (seventh layer). The weights used to inflate from the latent layer to the fifth and the fifth to sixth are tied to those used to compress from the second layer to the third and from the first layer to the second.

figure

*3) Evaluating results:*

*a) Evaluation method:* We will first try to to pinpoint a threshold on the distance between two vectors which separates pairs of vectors labeled to the same locutor and labeled to two separate locutors. To this end, we will compute the distance (discussed in IV-A) between every pair of vectors labeled to the same locutor. The maximum of those distances is then computed and chosen as our **threshold**.

We know by construction that every pair labeled to the same locutor is below the chosen threshold. To evaluate the relevance of the studied representation we then count the number of pairs of unmatched vectors whose corresponding distance is below our threshold. and would be misclassified if our threshold is was used as a classification method.

*b) Comparing to i-vectors:* To substantiate our claim that i-vector extraction does not lead to a clustering of same speaker vectors, we will apply our method to both our extracted intermmediate vectors and standard *i-vectors*. Those intermediate representations will be extracted from source files using the tool ALIZE ([6])