A DIAGONALIZED NEWTON ALGORITHM FOR NON-NEGATIVE SPARSE CODING

Hugo Van hamme

Department of Electrical Engineering – ESAT – KU Leuven, Leuven, Belgium

ABSTRACT

Signal models where non-negative vector data are represented by a sparse linear combination of non-negative basis vectors have attracted much attention in problems including image classification, document topic modeling, sound source segregation and robust speech recognition. In this paper, an iterative algorithm based on Newton updates to minimize the Kullback-Leibler divergence between data and model is proposed. It finds the sparse activation weights of the basis vectors more efficiently than the expectation-maximization (EM) algorithm. To avoid the computational burden of a matrix inversion, a diagonal approximation is made and therefore the algorithm is called diagonal Newton Algorithm (DNA). It is several times faster than EM, especially for undercomplete problems. But DNA also performs surprisingly well on overcomplete problems.

Index Terms— sparse coding, non-negative matrix factorization, source separation, Newton method, Kullback Leibler divergence, vocabulary acquisition.

1. INTRODUCTION

Many proposed solutions to problems in speech, image or text processing model non-negative vector data by a sparse linear combination of non-negative basis vectors with non-negative activation weights. Examples in audio processing [1] include separating sound sources from single-channel mixtures [2][3], dereverberation [4], music transcription [5], noise-robust speech recognition [6][7], vocabulary acquisition and recognition [8] up to bird phrase classification [9]. The non-negative basis vectors can be learned from data using non-negative matrix factorization (NMF) or they can be sampled from data (exemplars) or they can be constructed from insights about the data. Though this term is mostly used in conjunction with the exemplar approach, the set of non-negative basis vectors will be called the *dictionary*. Once the dictionary is chosen, the problem becomes to estimate the non-negative *weights* or *activations* of the dictionary elements, a problem that will be referred to as non-negative sparse coding (NSC) in this paper. Especially in cases where the dimension of the dictionary elements is larger than the number of dictionary elements, the activations will tend to be sparse. To increase sparsity, a regularization term is added in the NSC problem.

The linear combination will only approximate the data to be analyzed and hence a metric is required to quantify the quality of this approximation. In this paper, the Kullback-Leibler divergence (KLD) will be used for which the multiplicative update (or expectation-maximization, EM) rule of [10] applies. Algorithms for the more generic formulation of α,β-divergence can be found in [11].

The EM update converges slowly and this is often a practical concern. The goal of this paper is to increase the convergence speed by using second order (Newton) updates. However, for multivariate problems like NSC, this typically results in a matrix inversion, which destructs the benefit of faster convergence. There is prior work on second order updates for the least squares cost function, e.g. [12][13][14], but none uses a diagonal approximation without line search. The method in [15] is also fit for the KLD but still involves QR-decomposition and Levenberg regularization for convergence. In this paper, a diagonal approximation not involving line search is made instead, making the update computationally attractive, even for high-dimensional problems. The resulting algorithm is named *diagonal Newton algorithm* (DNA). To the best of the author's knowledge, this approach has never been explored before.

Below, the NSC problem is first formulated in section 2 and the baseline EM solution is explained in section 3. DNA is then motivated in sections 4 and 5 and evaluated on practical speech processing problems section 6.

2. NSC FORMULATION

Let $\mathbf{W} \in \mathbb{R}^{N\times R}$ be the matrix containing R dictionary vectors, each of dimension N. In the present formulation, it is not assumed that the dictionary elements are normalized. In the absence of sparsity regularization, there is a computational benefit for normalizing the dictionary elements to unit ℓ_1-norm. Let $\mathbf{h} \in \mathbb{R}^R$ denote the weight vector with which a non-negative data vector $\mathbf{v} \in \mathbb{R}^N$ is modeled as $\mathbf{v} \approx \mathbf{W}\,\mathbf{h}$. All parameters are assumed to be non-negative: $w_{ij}\geq 0$, $h_i\geq 0$ and $v_i\geq 0$. In this work, the similarity between $\mathbf{v}$ and $\mathbf{z} = \mathbf{W}\,\mathbf{h}$ is measured by the extended Kullback-Leibler divergence

$$d(\mathbf{v},\mathbf{z}) = \sum_n v_n log\left(\frac{v_n}{z_n}\right) - \sum_n v_n + \sum_n z_n$$

Though non-negativity already leads to sparse $\mathbf{h}$, sparsity in $\mathbf{h}$ can be encouraged by *regularizing* $d(\mathbf{v},\mathbf{z})$ by penalizing large weights:

$$d_R(\mathbf{v}, \mathbf{W}\,\mathbf{h}) = d(\mathbf{v}, \mathbf{W}\,\mathbf{h}) + \sum_r \lambda_r\, h_r \qquad (1)$$

where $\lambda_r \geq 0$ are the regularization parameters. NSC is then formulated as

$$\hat{\boldsymbol{h}} = \arg\min_{\boldsymbol{h}'} d_R(\mathbf{v}, \mathbf{W}\,\mathbf{h}')$$

subject to $\mathbf{h}' \geq \mathbf{0}$, which is a convex optimization problem with a unique solution. Consider the r-th component of the minimizer of (1). It should either be a stationary point, i.e.

$$\frac{\partial d_R(\mathbf{v}, \mathbf{W}\,\mathbf{h})}{\partial h_r} = -\sum_i v_i \frac{w_{ir}}{(\mathbf{W}\,\mathbf{h})_i} + \sum_i w_{ir} + \lambda_r = 0$$

or it should be on the boundary of the non-negativity constraint, i.e.

$$h_r = 0$$

Notice that in the latter case $\partial d_R/\partial h_r$ must be positive, for if it were not, the cost (1) could be reduced for some nonzero h_r, which would contradict the assumption that the optimum is on the boundary. Hence the minimizer should satisfy for $r = 1 \ldots R$

$$\sum_i v_i \frac{w_{ir} h_r}{(\mathbf{W}\,\mathbf{h})_i} - h_r \left(\sum_i w_{ir} + \lambda_r \right) = 0 \tag{2}$$

To simplify the expressions below, the N-dimensional column vector function $\mathbf{q}(\mathbf{h})$ is defined as:

$$q_i(\mathbf{h}) = \frac{v_i}{(\mathbf{W}\,\mathbf{h})_i}$$

Hence, with $\mathbf{1}$ denoting a vector of ones of appropriate length and superscript t denoting matrix transpose, (2) becomes:

$$h_r \frac{(\mathbf{W}^t \mathbf{q})_r}{(\mathbf{W}^t \mathbf{1})_r + \lambda_r} - h_r = 0$$

Finally, define

$$a_r(\mathbf{h}) = \frac{\left(\mathbf{W}^t \mathbf{q}(\mathbf{h})\right)_r}{(\mathbf{W}^t \mathbf{1})_r + \lambda_r} - 1 \tag{3}$$

so any solution must satisfy

$$a_r(\mathbf{h})\, h_r = 0 \quad \text{for } r = 1 \ldots R \tag{4}$$

Summing (2) over r yields

$$\sum_r ((\mathbf{W}^t \mathbf{1})_r + \lambda_r) h_r = \sum_i v_i \tag{5}$$

which is satisfied for any guess $\mathbf{h}$ by renormalizing:

$$h_r \leftarrow\cdots h_r \frac{\mathbf{v}^t \mathbf{1}}{\sum_k ((\mathbf{W}^t \mathbf{1})_k + \lambda_k) h_k} \tag{6}$$

For a normalized estimate of $\mathbf{h}$, the cost function is expressed as

$$d_R(\mathbf{v}, \mathbf{W}\,\mathbf{h}) = \sum_i v_i \log q_i \tag{7}$$

3. EM UPDATES

Equation (4) does not have an analytical solution for $\mathbf{h}$, but adding h_r to both sides, a fixed point update can be attempted:

$$h_r \leftarrow\cdots h_r(1 + a_r) = h_r \frac{(\mathbf{W}^t \mathbf{q})_r}{(\mathbf{W}^t \mathbf{1})_r + \lambda_r} \tag{8}$$

This update is recognized as multiplicative updates for the KLD metric [10] which has also been shown to be a form of the Expectation-Maximization (EM) algorithm [16], for which non-increase of the convex cost function (1) is guaranteed at each update. When initializing the EM algorithm, one has to be careful to choose all $h_r \neq 0$ for the update has a *zero locking* property, i.e. it can never escape a point where any component is zero. Notice also that whatever $\mathbf{h}$ the EM-update is applied to, the updated estimate will automatically satisfy (5).

Update (8) has two fixed points: $h_r = 0$ and $a_r = 0$. In the former case, a_r must be negative, for a_r is the negative of the derivative which was already proven to be positive in this case.

EM updates often converge slowly and authors report requiring hundreds or even thousands of iterations to attain solution that have sufficiently converged for their application. Hence there is an interest in speeding up the convergence process.

4. NEWTON UPDATES

In Section 2 it was shown that all stationary points satisfy equation (4). Instead of solving it with the fixed point EM update (8), a Newton method is proposed. In general, let $\mathbf{g}(\mathbf{h})$ be an R-dimensional vector function of an R-dimensional variable $\mathbf{h}$. Newton's update then states:

$$\mathbf{h} \leftarrow\cdots \mathbf{h} - (\nabla \mathbf{g})^{-1} \mathbf{g}(\mathbf{h}) \tag{9}$$

with

$$(\nabla \mathbf{g})_{rl} = \frac{\partial g_r(\mathbf{h})}{\partial h_l} \tag{10}$$

Applied to equation (4):

$$(\nabla \mathbf{g})_{rl} = a_l \delta_{rl} - \frac{h_r}{(\mathbf{W}^t \mathbf{1})_r + \lambda_r} \sum_i \frac{v_i w_{ir} w_{il}}{(\mathbf{W}\mathbf{h})_i^2} \tag{11}$$

where δ_{rl} is Kronecker's delta.

To avoid the matrix inversion in update (9), the last term in equation (11) is diagonalized, which is tantamount to solving the r-th equation in (4) for h_r with all other components fixed. With

$$b_r(\mathbf{h}) = \frac{1}{(\mathbf{W}^t \mathbf{1})_r + \lambda_r} \sum_i v_i \frac{w_{ir}^2}{(\mathbf{W}\,\mathbf{h})_i^2} \tag{12}$$

which is always positive, an element-wise Newton update for $\mathbf{h}$ is obtained:

$$h_r \leftarrow\cdots h_r \frac{h_r b_r(\mathbf{h})}{h_r b_r(\mathbf{h}) - a_r(\mathbf{h})} \tag{13}$$

Notice that this update does not automatically satisfy (5), so updates should be followed by a renormalization (6).

Like for the EM-update, $h_r = 0$ and $a_r = 0$ are the only fixed points of update (13), which are now shown to be *locally* stable. In case the optimizer is at $h_r = 0$, we have shown before that a_r is negative, and update (13) will indeed decrease h_r. In a sufficiently small neighborhood of a point where the gradient vanishes, i.e. $a_r = 0$, update (13) will increase (decrease) h_r if (8) increases (decreases) its estimate. Since if (8) is stable, update (13) must be too.

However, this only guarantees local convergence for per-element updates and Newton methods are known to suffer from potentially small convergence regions. This also applies to update (13), which can indeed result in limit cycles in some cases. In the next subsections, two measures are taken to respectively increase the convergence region and to make the update globally stable.

4.1 Step size limitation

When a_r is positive, update (13) may not be well-behaved in the sense that its denominator can become negative or zero. Therefore, it is bounded below by a function with the same local behavior around zero:

$$\frac{h_r b_r(\mathbf{h})}{h_r b_r(\mathbf{h}) - a_r(\mathbf{h})} = \frac{1}{1 - \frac{a_r(\mathbf{h})}{h_r b_r(\mathbf{h})}} \geq 1 + \frac{a_r(\mathbf{h})}{h_r b_r(\mathbf{h})} \tag{14}$$

Hence, if $a_r \geq 0$, the following update is used:

$$h_r \leftarrow\cdots h_r\left(1+\frac{a_r(\mathbf{h})}{h_r b_r(\mathbf{h})}\right)=h_r+\frac{a_r(\mathbf{h})}{b_r(\mathbf{h})} \quad (15)$$

Both the multiplicative and additive form of this update can be used, since if h_r would converge to zero, a_r has to be negative and the update does not apply.

Finally, step sizes are further limited by flooring resp. ceiling the multiplicative gain applied to h_r in update (13) and (15) to $\varepsilon = 1$ resp. $\alpha > 1$.

4.2 Non-increase of the cost

Despite the measures taken in section 4.1, the divergence can still increase under the Newton update, though this is the exception rather than the rule. A very safe option is to compute the EM update additionally and compare the cost function value for both updates. If the EM update would be better, the Newton update is rejected and the EM update is taken instead. This will guarantee non-increase of the cost function.

5. DIAGONAL NEWTON ALGORITHM

With the elements explained and motivated above, the Diagonal Newton Algorithm (DNA) can now be proposed.

5.1 Algorithm

The details of the proposed algorithm are listed in Table 1. For comparison, the EM update requires only steps 1 through 4 to be performed, since its non-increase property does not require checking the cost. In practice, checking this condition for DNA is only required for the first few iterations. After M successive successes for the DNA algorithm, steps 4, 9 and 10 can be removed leading to computational gains.

In step 8, update (13) is used if $a_r < 0$ and (15) elsewhere.

input: $\mathbf{W}$, $\mathbf{v}$, $\boldsymbol{\lambda}$ and an initial guess for $\mathbf{h}$
precompute $\mathbf{W}^t\mathbf{1}+\boldsymbol{\lambda}$, $\mathbf{v}^t\mathbf{1}$ and $\mathbf{W}\odot\mathbf{W}$
set $m = 0$
repeat for a fixed number of iterations:

step	Compute	Eq.	Operations
1	$\mathbf{Wh}$		$O(NR)$
2	$\mathbf{q} = \mathbf{v}\varnothing(\mathbf{Wh})$		$O(N)$
3	$\mathbf{A}$	(3)	$O(NR)+O(R)$
4	if $m < M$ compute $\mathbf{h}_{EM}$	(8)	$O(R)$
5	$(\mathbf{Wh})^2$		$O(N)$
6	$\mathbf{v}\varnothing(\mathbf{Wh})^2$		$O(N)$
7	$\mathbf{B}$	(12)	$O(NR)+O(R)$
8	$\mathbf{h}_{DNA}$	(13)(15)	$O(2R)$
9	Normalize	(6)	$O(2R)$
9	if $m < M$ compute cost for $\mathbf{h}_{EM}$	(7)	$O((K+1)N)$
10	if $m < M$ compute cost for $\mathbf{h}_{DNA}$	(7)	$O((K+1)N)$
11	accept solution with lowest cost. If that is $\mathbf{h}_{DNA}$, increment m, else set $m = 0$.		

Table 1: Processing steps for EM (step 1-4) and DNA with their complexity. The symbols $\odot$ and $\varnothing$ denote element-wise multiplication and division respectively. K is the CPU cost for evaluation of a logarithm expressed in elementary operations.

5.2 Computational complexity

The complexity of each step is listed in Table 1. Here, a multiplication, an addition and a multiply/accumulate are all counted as one elementary operation. The evaluation of a logarithm costs K elementary operations. For the CPU used in the experimental section, K was measured to be close to 1. On other CPUs, a value of about 5 was found.

The complexity of EM is $O(2NR+N+2R)$ for each iteration. For DNA, this becomes $O(3NR+(2K+1)N+7R)$ when $m < M$, and $O(3NR+3N+7R)$ afterwards. Theoretically, once $m \geq M$, DNA will be more efficient than EM if the number of iterations can be reduced by a one third (assuming the terms in NR dominate).

5.3 Approximation

A reduction in the number of iterations will only be observed if the diagonal approximation of equation (11) holds, which expresses that the basis vectors (columns of $\mathbf{W}$) should be orthogonal when weighted with $v_i/(\mathbf{Wh})_i^2$. Given the non-negativity of $\mathbf{W}$, a diagonal structure is achieved when the rows of $\mathbf{W}$ contain a single dominant value, especially when the corresponding entry in the data $\mathbf{v}$ is large. This condition is more easily satisfied when $\mathbf{W}$ is obtained from an NMF (and hence $N > R$), i.e. when the $\mathbf{W}$ columns are the R corner points of a convex hull constructed to span the training as closely as possible. Hence, for these undercomplete problems, a good performance is expected.

However, for NSC problems where the columns of $\mathbf{W}$ are obtained from potentially similar samples as will be the case in section 6.2,, orthogonality is questionable and is therefore included in the experimental verification.

6. EXPERIMENTS

In this section the DNA is tested on several data sets. M is set to 5, $\varepsilon = 0.01$ and $\alpha = 5$, irrespective of the matrix dimensions. Both EM and DNA are initialized from $\mathbf{W}^t\mathbf{v}$. Timing measurements are obtained on a quad-core AMD™ Opteron 8356 processor and all code is written in MATLAB™ using matrix/vector operation where possible. Software is publically available from http://www.esat.kuleuven.be/psi/spraak/downloads.

6.1 Vocabulary acquisition data

The first test case is one with a very "skinny" $\mathbf{W}$, i.e. it has dimensions $N = 165000$ and $R = 12$ and occurs in a task of vocabulary acquisition and subsequent recognition [8]. $\mathbf{W}$ is learned with NMF from training data. The data $\mathbf{v}$ are obtained by counting the number of times discrete events occur in an utterance. The physical meaning of the basis columns in $\mathbf{W}$ is the occurrence probability of the discrete events in each of the R words the utterances are composed of. The sparsity parameter $\lambda_r = 0$ for all r.

To examine the convergence, we perform a reference experiment where the EM update is run for 20000 iterations. Figure 1 then shows the absolute difference between these reference activation weights and the solutions found by both algorithms at each iteration, obviously showing superiority of DNA. This metric is preferred over the divergence, since it converges to zero. Moreover, in all evaluation cases, the divergence for DNA is always lower than for EM for an equal number of iterations.

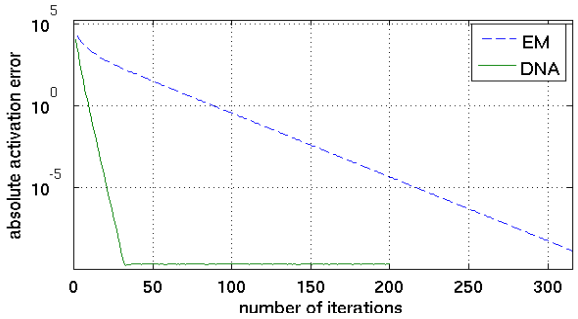


Figure 1: convergence of EM and DNA on the word acquisition data set with a **W** of size 165000 × 12.

6.2 Exemplar matching on spectra

The NSC problem in this task arises from matching sparse linear combinations of exemplars of root-compressed MEL-scale spectrograms of subword units (half digits in this case) to new data [17]. Exemplars of equal length and subword class are grouped in a dictionary which, depending on the exemplar length and the number of exemplars, can turn out to be undercomplete or overcomplete. Since all columns of **W** are alike, orthogonality is a questionable assumption. DNA and EM were compared on various problems of which the ones with largest *NR* ranged from 119 × 248 to 493 × 104. Each of the problems is solved for 100 data vectors using the sparsity regularization that emerges in [17].

Figure 2 shows the sum of the weight vector absolute errors over the 100 different data vectors along the utterance to be decoded for the 493 × 104 task. In 15 iterations, DNA reaches the same error as EM in 200 iterations. It is then 5.5 times faster. On the overcomplete 119 × 248 task (Figure 3), EM is particularly slow. DNA needs 45 iterations to achieve the same activation error as EM achieves in 1000 iterations and is 13 times faster on the hardware mentioned above.

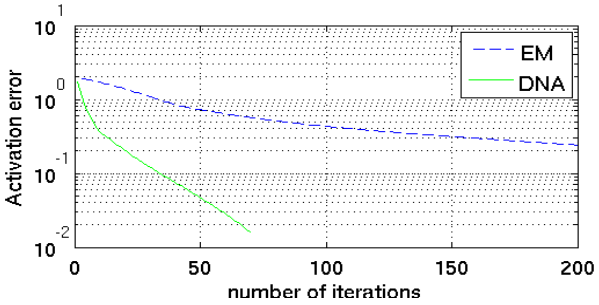


Figure 2: convergence of EM and DNA on the exemplar matching data set with a **W** of size 493 × 104.

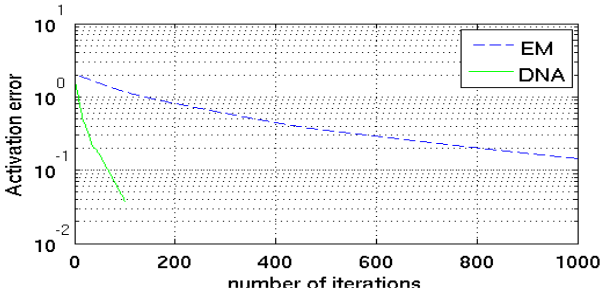


Figure 3: convergence of EM and DNA on the exemplar matching data set for an overcomplete 119 × 248 task.

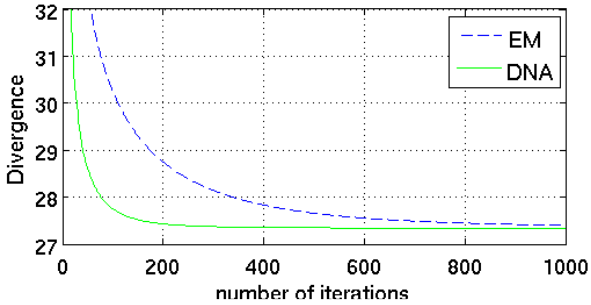


Figure 4: convergence of EM and DNA on a large overcomplete task of 690 × 14023

6.3 Robust speech recognition data

The largest case considered arises from sparse representations for feature enhancement in the context of speech recognition [6]. 7000 exemplars of Mel-scaled spectrograms with 23 channels of fixed length of 30 frames are extracted from speech and 7000 from noise data. Additionally, 23 single-channel stationary noise exemplars are added, yielding a **W** of size 690 × 14023. All dictionary elements are ℓ_2-normalized and all have the same sparsity regularization constant as dictated by the application. In this test, the activation weights for 315 test spectrograms extracted from noisy data are computed. Due to the large matrix size, convergence is slower and *M* is set equal to the number of iterations. For the same reason, it is expensive to compute a reference solution and therefore divergence is reported in Figure 4. To achieve the same divergence, DNA needs less than a quarter of the number of iterations required by EM and is about twice faster in practice.

7. CONCLUSIONS

Models based on sparse non-negative representations have shown excellent performance in many problems in speech, image and text processing. In many cases, the computational cost of these methods is a practical hurdle. The DNA algorithm can offer a solution to speed up the computation of activation weights in these NSC problems. Thanks to the diagonal approximation, matrix inversion is avoided resulting in an algorithm that is simple to implement and that scales well to large matrix sizes. To guarantee the non-increase of the cost function, a combination with the EM update is proposed. The computational cost of an iteration of DNA is larger than for EM, but this cost is won back by the increased convergence speed. Especially for skinny NMF problems ($R = N$) for which the diagonal approximation is expected to hold well, the convergence is much faster than for the EM algorithm. It is exactly in these undercomplete NSC problems that the main application of DNA is situated. But even for large overcomplete problems found in exemplar-based modeling computational gains are observed.

DNA has been found to be more efficient than EM on the speech processing problems studied in this paper. In future work, this evaluation will be extended with additional test cases originating from practical problems. DNA also offers potential in NMF problems where updates of activations and bases alternate.

8. ACKNOWLEDGEMENTS

This work is supported by KU Leuven grant OT/09/028 (VASI) and European Commission FP7-PEOPLE-ITN-2008 (BBfor2).

9. REFERENCES

[1] M. D. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, and M. E. Davies, "Sparse representations in audio & music: from coding to source separation," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 995–1005, 2009.

[2] P. Smaragdis, B. Raj, and M.V. Shashanka, "Supervised and Semi-Supervised Separation of Sounds from Single-Channel Mixtures," *in proceedings of the 7th International Conference on Independent Component Analysis and Signal Separation*. London, UK. September 2007.

[3] T. Virtanen, "Separation of Sound Sources by Convolutive Sparse Coding," *in proceedings of the ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, SAPA, Jeju, Korea, October 2004

[4] R. Singh, B. Raj, and P. Smaragdis, "Latent-variable decomposition based dereverberation of monaural and multi-channel signals," *in proceedings IEEE International Conference on Audio and Speech Signal Processing*, Dallas, TX, USA March 2010

[5] J. Paulus, and T. Virtanen, "Drum Transcription with Non-negative Spectrogram Factorisation," *in proc. 13th European Signal Processing Conference* Antalaya, Turkey, 2005

[6] J. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 19, no. 7, 2011.

[7] T. N. Sainath, B. Ramabhadran, D. Nahamoo, D. Kanevsky , D. Van Compernolle, K. Demuynck , J. F. Gemmeke , J. R. Bellegarda, and S. Sundaram, "Exemplar-Based Processing for Speech Recognition," *IEEE Signal Processing Magazine*, 2012.

[8] H. Van hamme, "HAC-models: a Novel Approach to Continuous Speech Recognition," In Proc. International Conference on Spoken Language Processing, pages 2554-2557, Brisbane, Australia, September 2008.

[9] L.N. Tan, K. Haewtip, M.L. Cody, C.E. Taylor and A. Alwan, "Evaluation of a Sparse Representation-Based Classifier for Bird Phrase Classificatio under Limited Data Conditions," *in proc. Interspeech*, Portland, USA, September 2012.

[10] D. Lee, and H. Seung, "Algorithms for non-negative matrix factorization," *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2001.

[11] A. Cichocki, S. Cruces, and S.-I. Amari, "Generalized Alpha-Beta Divergences and Their Application to Robust Nonnegative Matrix Factorization," *Entropy*, vol. 13, pages. 134-170, 2011; doi:10.3390/e13010134

[12] S. Bellavia, M. Macconi, and B. Morini, "An interior point Newton-like method for nonnegative least-squares problems with degenerate solution," *Numerical Linear Algebra with Applications*, vol. 13, no. 10, pp. 825-846, December 2006.

[13] Y. Zheng, and Q. Zhang, "Damped Newton based Iterative Non-negative Matrix Factorization for Intelligent Wood Defects Detection," *Journal of software*, vol. 5, no. 8, pp. 899-906, August 2010.

[14] P. Gong, and C. Zhang, "Efficient Nonnegative Matrix Factorization via projected Newton method", *Pattern Recognition*, vol. 45, no. 9, pp. 3557-3565, September 2012.

[15] R. Zdunek and A. Cichocki, "Non-Negative Matrix Factorization with Quasi-Newton Optimization," *Lecture Notes in Computer Science, Artificial Intelligence and Soft Computing* 4029, pp. 870-879, 2006

[16] E. Gaussier, and C. Goutte, "Relation between plsa and nmf and implications," *in SIGIR*, pp. 601–602, 2005

[17] E. Yilmaz, D. Van Compernolle, and H. Van hamme, "Combining Exemplar-based Matching and Exemplar-based Sparse Representations of Speech", *Symposium on Machine Learning in Speech and Language Processing (MLSLP),* Portland, USA, September 2012