Supplementary Materials to "On the Use of Sparse Filtering for Covariate Shift Adaptation"

Fabio Massimo Zennaro[1]**, Ke Chen**[2]

[1]Department of Informatics, University of Oslo, Oslo, Norway

[2]Department of Computer Science, The University of Manchester, Manchester, UK

A Proofs

A.1 Proof of Proposition 1

Proposition 1 *The sub-domain $\mathbb{Z}$ of the representations $\mathbf{z}_i$ learned by SF is* $[0, 1]^L$.

Proof. Let $\mathbf{x}_1 \in \mathbb{X}$ be a generic data sample to be processed through SF. Let the matrix $\tilde{\mathbf{F}}$ be the output of the ℓ_2-normalization along the rows of the SF algorithm, that is:

$$\tilde{\mathbf{F}} = \ell_{2,row} \left(|\mathbf{W}\mathbf{X}| \right).$$

The final output of the SF algorithm is then:

$$\mathbf{z}_1 = \frac{\tilde{f}_{1,j}}{\sqrt{\sum_{j=1}^{L}\left(\tilde{f}_{1,j}\right)^2}}.$$

Every feature $z_{1,j}$ is given by the feature value $\tilde{f}_{1,j}$ normalized by the ℓ_2-norm of $\tilde{\mathbf{f}}_1$. Therefore, it follows that each feature $z_{1,j}$ is bounded within $[0,1]$. Consequently, the representation $\mathbf{z}_1$ is bounded within the hyper-cube $[0,1]^L$. ■

A.2 Proof of Proposition 2

Proposition 2 *For each learned feature $\mathbf{z}_{\cdot,j}$, the SF algorithm bounds $E\left[Z_{\cdot,j}\right] \in [\epsilon, 1]$ and $Var\left[Z_{\cdot,j}\right] \in [0, 1-\epsilon^2]$, where $\epsilon > 0$ is an arbitrarily small positive value defined in the non-linearity of SF. Moreover, if learned representations are $[1,k]$-sparse in population and lifetime, and $\epsilon \to 0$, then we have the bounds $E\left[Z_{\cdot,j}\right] \in \left[\frac{1}{N} + \mathcal{O}\left(\epsilon\right), \frac{k}{N} + \mathcal{O}\left(\epsilon\right)\right]$ and $Var\left[Z_{\cdot,j}\right] \in \left[\frac{N-k^2}{N^2} - \mathcal{O}\left(\epsilon\right), \frac{Nk-1}{N^2} - \mathcal{O}\left(\epsilon\right)\right]$.*

Proof. The proof of this proposition is based on the following logical steps: (a) re-statement of the basic properties of the learned representations; (b) estimation of the expected value of the distribution of a learned feature (with and without the assumption of k-sparsity); (c) estimation of the second moment of the distribution of a learned feature (with and without the assumption of k-sparsity); (d) estimation of the variance of the distribution of a learned feature (with and without the assumption of k-sparsity).

(a) Let us consider the ℓ_2-normalization steps defining $\tilde{\mathbf{F}}$ and $\mathbf{Z}$. These transformations have two main effects: they constrain all the values in $\tilde{\mathbf{F}}$ and $\mathbf{Z}$ to be within $[0,1]$; and, they force features or samples to have a square total activation of 1. Formally:

$$1 \leqslant \tilde{f}_{i,j} \leqslant 0,\ 1 \leqslant z_{i,j} \leqslant 0,\ \ \forall 1 \leqslant j \leqslant L,\ 1 \leqslant i \leqslant N,$$

$$\sum_{i=1}^{N} \left(\tilde{f}_{i,j}\right)^2 = 1, \;\; 1 \leqslant j \leqslant L$$

$$\sum_{j=1}^{L} \left(z_{i,j}\right)^2 = 1, \;\; 1 \leqslant i \leqslant N.$$

(b) Let us now consider a given feature and, for clarity, let us denote this fixed feature as $\bar{j}$ to underline the fact that it is not going to change in the following analysis. We can now analyse the distribution of $z_{\cdot,\bar{j}}$ by considering its main statistical moments. Let us start by analysing the expected value of the random variable $Z_{\cdot,\bar{j}}$:

$$E\left[Z_{\cdot,\bar{j}}\right] \hat{=} \frac{1}{N} \sum_{i=1}^{N} z_{i,\bar{j}}.$$

After normalizing along the column, the quantity $\sum_{i=1}^{N} z_{i,\bar{j}}$ is not rigidly constrained. The value of the feature $z_{i,\bar{j}}$ can range between ϵ (if the feature $\bar{j}$ happens to be inactive for the sample i) and 1 (if the feature is the only active feature for the sample i). Therefore, the expected value can be bound in:

$$\epsilon \leqslant E\left[Z_{\cdot,\bar{j}}\right] \leqslant 1.$$

Let us now make the assumption that the feature $z_{\cdot,\bar{j}}$ is at most k-sparse, with $1 < k < L$, that is, it is active on a number k of samples, with k greater than 1 and smaller than L. This assumption is justified by considering the properties of population sparsity and lifetime sparsity of SF (Ngiam, Chen, Bhaskar, Koh, & Ng, 2011). In this case, the new expected value can be evaluate as:

$$E\left[Z_{\cdot,\bar{j}}\right] = \int E\left[Z_{\cdot,\bar{j}} | Z_{\cdot,\bar{j}} \text{ is sparse}\right] P\left(Z_{\cdot,\bar{j}} \text{ is sparse}\right) dZ_{\cdot,\bar{j}}$$

under the assumption that $P\left(Z_{\cdot,\bar{j}} \text{ is sparse}\right) = 1$. The new expected value can then be bound in:

$$\frac{1 + (N-1)\epsilon}{N} \leqslant E\left[Z_{\cdot,\bar{j}}\right] \leqslant \frac{k + (N-k)\epsilon}{N}.$$

Moreover, if we assume that $\epsilon \to 0$, then the final bound can be re-written as:

$$\frac{1}{N} + \mathcal{O}\left(\epsilon\right) \leqslant E\left[Z_{\cdot,\bar{j}}\right] \leqslant \frac{k}{N} + \mathcal{O}\left(\epsilon\right).$$

This proves the first part of our statement.

(c) Let us now consider the estimation of the second moment:

$$M_2\left[Z_{\cdot,\bar{j}}\right] = E\left[\left(Z_{\cdot,\bar{j}}\right)^2\right] \hat{=} \frac{1}{N}\sum_{i=1}^{N} z_{i,\bar{j}}^2.$$

For the same reason we gave above about the admissible values for the feature $z_{\cdot,\bar{j}}$, the second moment can be bound in:

$$\epsilon^2 \leqslant M_2\left[Z_{\cdot,\bar{j}}\right] \leqslant 1.$$

Under the assumption of k-sparsity of $z_{\cdot,\bar{j}}$, we can get the tighter bounds:

$$\frac{1+(N-1)\epsilon^2}{N} \leqslant M_2\left[Z_{\cdot,\bar{j}}\right] \leqslant \frac{k+(N-k)\epsilon^2}{N}.$$

(d) Finally, let us consider the estimation of the variance of $Z_{\cdot,\bar{j}}$:

$$Var\left[Z_{\cdot,\bar{j}}\right] = E\left[{Z_{\cdot,\bar{j}}}^2\right] - E\left[Z_{\cdot,\bar{j}}\right]^2.$$

Again, using the values we computed for the second moment and the expected value, we can define the following bounds for the variance:

$$\begin{aligned} \epsilon^2 - 1^2 &\leqslant Var\left[Z_{\cdot,\bar{j}}\right] \leqslant & 1-\epsilon^2 \\ 0 &\leqslant Var\left[Z_{\cdot,\bar{j}}\right] \leqslant & 1-\epsilon^2. \end{aligned}$$

Under the assumption of k-sparsity we can recompute the bounds:

$$\begin{aligned} Var\left[Z_{\cdot,\bar{j}}\right] &\geqslant \frac{1+(N-1)\epsilon^2}{N} - \left(\frac{k+(N-k)\epsilon}{N}\right)^2 \\ &\geqslant \frac{N(1-\epsilon^2+2k\epsilon^2-2k\epsilon)-k^2(1+\epsilon^2-2\epsilon)}{N^2}, \end{aligned}$$

$$\begin{aligned} Var\left[Z_{\cdot,\bar{j}}\right] &\leqslant \frac{k+(N-k)\epsilon^2}{N} - \left(\frac{1+(N-1)\epsilon}{N}\right)^2 \\ &\leqslant \frac{N(k-k\epsilon^2+2\epsilon^2-2\epsilon)-1-\epsilon^2+2\epsilon}{N^2}. \end{aligned}$$

If we take $\epsilon \to 0$, then the bounds of the variance are:

$$\frac{N-k^2}{N^2} - \mathcal{O}\left(\epsilon\right) \leqslant Var\left[Z_{\cdot,\bar{j}}\right] \leqslant \frac{Nk-1}{N^2} - \mathcal{O}\left(\epsilon\right).$$

This proves the last part of our statement. ■

A.3 Proof of Theorem 3

Theorem 3 *Let* $\mathbf{x}_1 \in \mathbb{R}^M$ *be a point in the original space and let* $\mathbf{z}_1 \in \mathbb{R}^L$ *be its corresponding representation learned by PSF. Then there is an infinite set of points* $\mathbf{x}_i \in \mathbb{R}^M$ *that map onto the same representation* $\mathbf{z}_1$*. The set of the points* $\mathbf{x}_i \in \mathbb{R}^M$ *built from* $\mathbf{x}_1$ *with period* $\mathbf{W}^g\boldsymbol{\kappa}\pi$*, where* $\mathbf{W}^g$ *is the generalized inverse of the weight matrix of PSF and* $\boldsymbol{\kappa}$ *is a vector of integer constants in* $\mathbb{Z}$*, is included in this set.*

Proof. The proof of this theorem is based on identifying the periodic filters defined by PSF in the original space and showing that points falling within these filters are mapped onto identical representations. The proof makes the following logical steps: (a) rigorous definition of the PSF computation; (b-e) back-computation through all the steps of PSF up to the input (ℓ_2-normalization along the columns, ℓ_2-normalization along the rows, non-linearity, linear projection).

(a) Let us consider two points in the original space $\mathbb{R}^M$:

$$\begin{aligned} \mathbf{x}_1 &= \begin{bmatrix} x_{1,1} & x_{1,2} & \ldots & x_{1,M} \end{bmatrix}^\top \\ \mathbf{x}_2 &= \begin{bmatrix} x_{2,1} & x_{2,2} & \ldots & x_{2,M} \end{bmatrix}^\top, \end{aligned}$$

and their corresponding representations in the learned space $\mathbb{R}^L$ defined by PSF:

$$\begin{aligned} \mathbf{z}_1 &= \left[\begin{array}{cccc} z_{1,1} & z_{1,2} & \dots & z_{1,L} \end{array} \right]^\top \\ \mathbf{z}_2 &= \left[\begin{array}{cccc} z_{2,1} & z_{2,2} & \dots & z_{2,L} \end{array} \right]^\top . \end{aligned}$$

Let us also consider a version of PSF implemented using a strictly positive element-wise sine function: $PSF\left(\mathbf{x}_i\right) = \ell_{2,col}\left(\ell_{2,row}\left(1 + \epsilon + \sin\left(\mathbf{W}\mathbf{x}_1\right)\right)\right)$.

Finally, let us assume that the two learned representations are identical, that is $\mathbf{z}_1 = \mathbf{z}_2$.

(b) By definition of PSF, $\mathbf{z}_1 = \mathbf{z}_2$ implies:

$$\begin{aligned} \ell_{2,col}\left(\tilde{\mathbf{f}}_1\right) &= \ell_{2,col}\left(\tilde{\mathbf{f}}_2\right) \\ \frac{\tilde{f}_{1,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{1,j}^2}} &= \frac{\tilde{f}_{2,j}}{\sqrt{\sum_{j=1}^{L} \tilde{f}_{2,j}^2}}, \end{aligned}$$

where $\tilde{\mathbf{f}}_i = \left[\begin{array}{cccc} \tilde{f}_{i,1} & \tilde{f}_{i,2} & \dots & \tilde{f}_{i,L} \end{array} \right]^\top$ is the intermediate output of PSF which is defined as $\tilde{\mathbf{F}} = \ell_{2,row}\left(1 + \epsilon + \sin\left(\mathbf{W}\mathbf{x}_1\right)\right)$. Now, for the ℓ_2-normalizations along the columns to be equal, it must hold that:

$$\left[\begin{array}{cccc} \frac{\tilde{f}_{1,1}}{d_1} & \frac{\tilde{f}_{1,2}}{d_1} & \dots & \frac{\tilde{f}_{1,L}}{d_1} \end{array} \right]^\top = \left[\begin{array}{cccc} \frac{\tilde{f}_{2,1}}{d_2} & \frac{\tilde{f}_{2,2}}{d_2} & \dots & \frac{\tilde{f}_{2,L}}{d_2} \end{array} \right]^\top ,$$

where $d_i = \sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}$ is a sample-dependent scaling factor. Therefore, it follows that $\mathbf{z}_1 = \mathbf{z}_2$ if and only if $\tilde{\mathbf{f}}_1 = \lambda \tilde{\mathbf{f}}_2$, for $\lambda \in \mathbb{R}$.

(c) By definition of PSF, $\tilde{\mathbf{f}}_1 = \lambda \tilde{\mathbf{f}}_2$ implies:

$$\begin{aligned} \ell_{2,row}\left(\mathbf{f}_1\right) &= \lambda \ell_{2,row}\left(\mathbf{f}_2\right) \\ \frac{f_{1,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}} &= \lambda \frac{f_{2,j}}{\sqrt{\sum_{i=1}^{N} f_{i,j}^2}}, \end{aligned}$$

where $\mathbf{f}_i = \begin{bmatrix} f_{i,1} & f_{i,2} & \dots & f_{i,L} \end{bmatrix}^\top$ is the intermediate output of PSF defined as $\mathbf{F} = 1 + \epsilon + \sin(\mathbf{W}\mathbf{x}_1)$. Now, for the ℓ_2-normalizations along the rows to be equal, it must hold that:

$$\begin{bmatrix} \frac{f_{1,1}}{t_1} & \frac{f_{1,2}}{t_2} & \dots & \frac{f_{1,L}}{t_L} \end{bmatrix}^\top = \lambda \begin{bmatrix} \frac{f_{2,1}}{t_1} & \frac{f_{2,2}}{t_2} & \dots & \frac{f_{2,L}}{t_L} \end{bmatrix}^\top,$$

where $t_j = \sqrt{\sum_{i=1}^{N} f_{i,j}^2}$ is a feature-dependent scaling factor. Therefore, it follows that $\tilde{\mathbf{f}}_1 = \lambda \tilde{\mathbf{f}}_2$ if and only if $\mathbf{f}_1 = \lambda \mathbf{f}_2$.

(d) By definition of PSF, $\mathbf{f}_1 = \lambda \mathbf{f}_2$ implies:

$$1 + \epsilon + \sin(\mathbf{h}_1) = \lambda (1 + \epsilon + \sin(\mathbf{h}_2)),$$

where $\mathbf{h}_i = \begin{bmatrix} h_{i,1} & h_{i,2} & \dots & h_{i,L} \end{bmatrix}^\top$ is the intermediate output of PSF defined as $\mathbf{H} = \mathbf{W}\mathbf{X}$. Now, in order to prove our statement about the set of points $\mathbf{x}_i \in \mathbb{R}^M$ built from $\mathbf{x}_1$ with period $\mathbf{W}^g \boldsymbol{\kappa}\pi$, let us consider the case where $\lambda = 1$:

$$\begin{aligned} 1 + \epsilon + \sin(\mathbf{h}_1) &= 1 + \epsilon + \sin(\mathbf{h}_2) \\ \sin(\mathbf{h}_1) &= \sin(\mathbf{h}_2). \end{aligned}$$

For the applications of the sinusoidal function to be equal, it must hold that:

$$\begin{aligned} \sin(\mathbf{h}_1) &= \sin(\mathbf{h}_2) \\ \mathbf{h}_1 &= (-1)^{\boldsymbol{\kappa}} \arcsin(\sin(\mathbf{h}_2)) + \boldsymbol{\kappa}\pi \\ \mathbf{h}_1 &= (-1)^{\boldsymbol{\kappa}} \mathbf{h}_2 + \boldsymbol{\kappa}\pi, \end{aligned}$$

where $\boldsymbol{\kappa}$ is a vector of feature-dependent periodic factors in $\mathbb{Z}$.

(e) By definition of PSF, $\mathbf{h}_1 = (-1)^{\boldsymbol{\kappa}} \mathbf{h}_2 + \boldsymbol{\kappa}\pi$ implies:

$$\begin{aligned} \mathbf{W}\mathbf{x}_1 &= (-1)^{\boldsymbol{\kappa}} \mathbf{W}\mathbf{x}_2 + \boldsymbol{\kappa}\pi \\ \mathbf{x}_1 &= (-1)^{\boldsymbol{\kappa}} \mathbf{x}_2 + \mathbf{W}^g \boldsymbol{\kappa}\pi, \end{aligned}$$

where $\mathbf{W}^g$ is the generalized inverse of $\mathbf{W}$. Thus, there are infinite points $\mathbf{x}_i \in \mathbb{R}^M$ built from $\mathbf{x}_1$ with period $\mathbf{W}^g \boldsymbol{\kappa}\pi$ that maps onto the same representation $\mathbf{z}_1$. ■

B Pseudo-Code

B.1 Pseudo-Code for the Derivative of PSF

Algorithm 1 lists the pseudo-code for the gradient descent on the PSF loss function.

Algorithm 1 Derivative for PSF

```
    **Input:** input data $\mathbf{X}$; weight matrix $\mathbf{W}$; PSF output $\mathbf{Z}$; PSF intermediate representations $\mathbf{H}$, $\mathbf{F}$, $\tilde{\mathbf{F}}$.
    **Hyper-params:** lambda vector $\lambda$.

$\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \leftarrow \frac{\sqrt{\sum_{j=1}^{L} \tilde{f}_{i,j}^2} - z_{i,j} \cdot \sum_{j=1}^{L} \tilde{f}_{i,j}}{\sum_{j=1}^{L} \tilde{f}_{i,j}^2}$
$\left[\frac{\partial \mathbf{Z}}{\partial \mathbf{F}}\right]_{i,j} \leftarrow \frac{\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \sqrt{\sum_{i=1}^{N} f_{i,j}^2} - \tilde{f}_{i,j} \cdot \sum_{i=1}^{N} \left(\left[\frac{\partial \mathbf{Z}}{\partial \tilde{\mathbf{F}}}\right]_{i,j} \cdot f_{i,j}\right)}{\sum_{i=1}^{N} f_{i,j}^2}$
$\frac{\partial \mathbf{Z}}{\partial \mathbf{H}} \leftarrow \frac{\partial \mathbf{Z}}{\partial \mathbf{F}} \cdot \cos \mathbf{H}$
$\frac{\partial \mathbf{Z}}{\partial \mathbf{W}} \leftarrow \lambda \frac{\partial \mathbf{Z}}{\partial \mathbf{H}} \cdot \mathbf{X}$
    **return** $\frac{\partial \mathbf{Z}}{\partial \mathbf{W}}$
```

C Experimental Validation

The eight CSA classification systems used in our experiments were implemented and configured as follows:

(i) *SVM (without CSA)*: SVM is implemented using the *scikit* implementation[1].

(ii) *SF+SVM*: SF is implemented using the code provided by Ngiam[2] (Ngiam et al., 2011). SVM is implemented as for system (i).

(iii) *PSF+SVM*: PSF is implemented using our own code[3]. SVM is implemented as for system (i).

(iv) *SFsine+SVM*: SFsine is implemented using our own code[4]. SVM is implemented as for system (i).

(v) *SFlabel+SVM*: SFlabel is implemented using our own code[5]. SVM is implemented as for system (i).

(vi) *IW+LSPC*: the integrated IW+LSPC is implemented using the code provided by Hachiya[6] (Hachiya, Sugiyama, & Ueda, 2012).

(vii) *SSA+SVM*: SSA is implemented using the code provided by Fernando[7] (Fernando, Habrard, Sebban, & Tuytelaars, 2013). SVM is implemented using the *Matlab* implementation.

(viii) *DAE+SVM*: DAE is implemented using the code provided by Mitra[8]. SVM is implemented as for system (i).

[1] `http://scikit-learn.org/`

[2] `https://github.com/jngiam/sparseFiltering`

[3] `https://github.com/FMZennaro/PSF`

[4] `https://github.com/FMZennaro/PSF`

[5] `https://github.com/FMZennaro/PSF`

[6] `http://www.ms.k.u-tokyo.ac.jp/software.html#IWLSPC`

[7] `http://users.cecs.anu.edu.au/~basura/DA_SA/`

[8] `https://github.com/rajarsheem/libsdae`

C.1 Synthetic Data Experiments

Data Generation

The four synthetic datasets were generated as follows:

(a) *Radial dataset*: $\mathbf{X^{tr}}$ consists of 500 samples from $p\left(X^{tr}\right) \sim \mathcal{N}\left(\begin{bmatrix} 0.5 \\ 0 \end{bmatrix}; \begin{bmatrix} 0.2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 500 samples from $p\left(X^{tst}\right) \sim \mathcal{N}\left(\begin{bmatrix} -0.5 \\ 0 \end{bmatrix}; \begin{bmatrix} 0.2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tar}}$ consists of 250 samples from $p\left(X^{tst}\right)$; $p\left(Y|X\right)$ is described by the following deterministic function $f\left(\mathbf{x}_i\right) \begin{cases} 1 & \text{if} \quad |x_{i,1}| > |x_{i,2}| \\ 0 & \text{otherwise} \end{cases}$, which defines two cones centered on the *x*-axis.

(b) *Periodic dataset*: $\mathbf{X^{tr}}$ consists of 500 samples from $p\left(X^{tr}\right) \sim \mathcal{N}\left(\begin{bmatrix} 2\pi \\ 0 \end{bmatrix}; \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 500 samples from $p\left(X^{tst}\right) \sim \mathcal{N}\left(\begin{bmatrix} -2\pi \\ 0 \end{bmatrix}; \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$; $\mathbf{X^{tar}}$ consists of 250 samples from $p\left(X^{tst}\right)$; $p\left(Y|X\right)$ is described by the following deterministic function $f\left(\mathbf{x}_i\right) \begin{cases} 1 & \text{if} \quad \sin|x_{i,1}| > 0 \\ 0 & \text{otherwise} \end{cases}$, which defines a periodic pattern perpendicular to the *x*-axis.

(c) *Smooth dataset*: $\mathbf{X^{tr}}$ consists of 250 samples from $p\left(X^{tr1}\right) \sim \mathcal{N}\left(\begin{bmatrix} 2 \\ 3 \end{bmatrix}; \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\right)$

and 250 samples from $p\left(X^{tr2}\right) \sim \mathcal{N}\left(\begin{bmatrix} -2 \\ 3 \end{bmatrix}; \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\right)$; $\mathbf{X^{tst}}$ consists of 250 samples from $p\left(X^{tst1}\right) \sim \mathcal{N}\left(\begin{bmatrix} 3 \\ -1 \end{bmatrix}; \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$ and 250 samples from $p\left(X^{tst2}\right) \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ -1 \end{bmatrix}; \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$; $\mathbf{X^{tar}}$ consists of 125 samples from $p\left(X^{tst1}\right)$ and 125 samples from $p\left(X^{tst2}\right)$; $p\left(Y|X\right)$ is described by the deterministic function $f\left(\mathbf{x}_i\right)\begin{cases} 1 & \text{if } \frac{1+\tanh(x_{i,1}+\min(0,x_{i,2}))}{2} > 0.5 \\ 0 & \text{otherwise} \end{cases}$. This dataset is the same as the one studied in Hachiya et al. (2012).

(d) *Diagonal dataset*: $\mathbf{X^{tr}}$ consists of 500 noisy samples taken along the diagonal of the first quadrant, $x_{i,1}^{\mathbf{tr}} \sim Unif\left([0,3]\right)$ and $x_{i,2}^{\mathbf{tr}} = x_{i,1}^{\mathbf{tr}} + \mathcal{N}\left(0, 0.2\right)$; $\mathbf{X^{tst}}$ consists of 500 noisy samples taken along the diagonal of the fourth quadrant, $x_{i,1}^{\mathbf{tst}} \sim Unif\left([-3,0]\right)$ and $x_{i,2}^{\mathbf{tst}} = x_{i,1}^{\mathbf{tst}} + \mathcal{N}\left(0, 0.2\right)$; $\mathbf{X^{tar}}$ consists of 250 noisy samples taken along the diagonal of the fourth quadrant as $\mathbf{X^{tst}}$; $p\left(Y|X\right)$ is described by the deterministic function

$$f\left(\mathbf{x}_i\right)\begin{cases} 1 & \text{if } |x_{i,2}| > 1.5 \\ 0 & \text{otherwise} \end{cases}.$$

Experimental Setup

The eight CSA classification systems are configured as follows:

(i) *SVM (with no CSA)*: a linear SVM is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$, using a fixed penalty $C = 1$.

(ii) *SF+SVM*: SF is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ for 500 iterations; we set the learned dimensionality to 2. SVM is trained as for system (i).

(iii) *PSF+SVM*: PSF is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}, \mathbf{Y^{tr}}\}$ for 500 iterations; we set the learned dimensionality to 2 (divided evenly between the two classes), the non-linearity to sine, and $\lambda = 10.0$. SVM is trained as for system (i).

(iv) *SFsine+SVM*: SFsine is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ for 500 iterations; we set the learned dimensionality to 2, the non-linearity to sine. SVM is trained as for system (i).

(v) *SFlabel+SVM*: SFlabel is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}, \mathbf{Y^{tr}}\}$ for 500 iterations; we set the learned dimensionality to 2 (divided evenly between the two classes), the non-linearity to absolute value, and $\lambda = 10.0$. SVM is trained as for system (i).

(vi) *IW+LSPC*: IW is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ using the pre-set *uLSIF* algorithm with 250 basis, candidate $\sigma = \{0.1, 0.2, 0.5, 1, 2, 3\}$ and candidate $\lambda = \{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10\}$. LSPC is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ with the same pre-set candidate $\sigma = \{0.1, 0.2, 0.5, 1, 2, 3\}$ and candidate $\lambda = \{0.1, 0.17, 0.32, 0.56, 1\}$.

(vii) *SSA+SVM*: SSA is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ using 2 PCA components. SVM is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$, using a penalty $C = 3$ to achieve the same baseline results as systems (i)-(iii).

(viii) *DAE+SVM*: DAE is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ for 10000 epochs with mini-batches of size 50 and learning rate of 0.001; we set the learned dimensionality to 1, the non-linearity to sigmoid, and the noise to Gaussian $\mathcal{N}\left(0, 0.1\right)$. SVM is trained as

Table 1: Accuracy change when using different CSA systems on the four synthetic data sets. Accuracy without the CSA algorithm to left of the arrow and accuracy with the CSA algorithm to the right of the arrow.

	SF+SVM	***PSF+SVM***	***IW+LSPC***	***SSA+SVM***	***DAE+SVM***
Radial	$0.342 \rightarrow 0.779 \pm 0.03$	$0.342 \rightarrow 0.473 \pm 0.06$	$0.336 \rightarrow 0.598 \pm 0.06$	$0.342 \rightarrow 0.342$	$0.342 \rightarrow 0.387 \pm 0.05$
Periodic	$0.488 \rightarrow 0.5 \pm 0.01$	$0.488 \rightarrow 0.619 \pm 0.02$	$0.488 \rightarrow 0.512 \pm 0.0$	$0.488 \rightarrow 0.488$	$0.488 \rightarrow 0.512 \pm 0.0$
Smooth	$0.894 \rightarrow 0.476 \pm 0.06$	$0.894 \rightarrow 0.538 \pm 0.04$	$0.865 \rightarrow 0.936 \pm 0.02$	$0.89 \rightarrow 0.89$	$0.894 \rightarrow 0.613 \pm 0.02$
Diagonal	$0.866 \rightarrow 0.584 \pm 0.02$	$0.866 \rightarrow 0.51 \pm 0.06$	$0.768 \rightarrow 0.786 \pm 0.01$	$0.86 \rightarrow 0.932$	$0.866 \rightarrow 0.514 \pm 0.0$

for system (i).

Further Results

Table 1 reports the accuracy of all the CSA classification systems before and after introducing CSA. The results show that, even if the baselines of the five CSA classification systems are not exactly the same, they are still comparable; therefore, it makes sense to compare the percentage change in accuracy when processing a given data set.

Table 2: Mapping of labels specific to each data set onto binary valence classes.

	Positive Valence	*Negative Valence*
EMODB	joy, neutral	anger, boredom, disgust, fear, sadness
DES	happiness, neutral, surprise	angry, sadness
VAM	$valence > 0$	$valence < 0$
eNT	joy, surprise	anger, disgust, fear, sadness

C.2 Real-World Data Experiments: Emotional Speech Data

Experimental Setup

All the recordings are pre-processed into standard feature representations using the open-source platform OpenSMILE[9]. For each 1-second sample we compute features on 2 *domains* (raw, delta), extracting 12 *descriptors* (12 Mel-frequency cepstrum coefficient) and computing 3 *statistical operators* (mean, standard deviation and range), for a total of 72 features. Labels are aligned along the valence dimension using a standard mapping in the ESR community (Schuller et al., 2010) as specified in Table 2.

All the samples are normalized per speaker, as in Schuller et al. (2010). Training and test data are upsampled using a simple re-sampling with repetition procedure, as in Zhang, Deng, and Schuller (2013).

In order to perform model selection over a reasonable set of values for the hyper-parameters of each algorithm, the six CSA classification systems are configured as follows:

[9] `http://audeering.com/technology/opensmile/`

Table 3: Best hyper-parameter configurations chosen by model selection.

	noCSA+SVM	***SF+SVM***	***PSF+SVM***
EMODB	SVM: $C = 7.5 \cdot 10^{-5}$	SF: $L = 50$ SVM: $C = 0.075$	PSF: $g = \cos()$ PSF: $L = 80$ PSF: $\lambda = 1.8$ SVM: $C = 7.5 \cdot 10^{-5}$
DES	SVM: $C = 5 \cdot 10^{-5}$	SF: $L = 120$ SVM: $C = 0.001$	PSF: $g = \sin()$ PSF: $L = 100$ PSF: $\lambda = 1.6$ SVM: $C = 1 \cdot 10^{-4}$
eNT	SVM: $C = 0.01$	SF: $L = 50$ SVM: $C = 1$	PSF: $g = \sin()$ PSF: $L = 120$ PSF: $\lambda = 1.8$ SVM: $C = 7.5 \cdot 10^{-5}$
	IW+LSPC	***SSA+SVC***	***DAE+SVM***
EMODB	LSPC: $\sigma = 3$ LSPC: $\lambda = 0.1$	SSA: $P = 50$ SVM: $C = 7.5 \cdot 10^{-5}$	DAE: $f, g = \text{sigm}()$ DAE: $L = 70$ DAE: $\sigma = 0.1$ DAE: $\eta = 0.005$ SVM: $C = 0.00025$
DES	LSPC: $\sigma = 3$ LSPC: $\lambda = 1$	SSA: $P = 50$ SVM: $C = 2.5 \cdot 10^{-4}$	DAE: $f, g = \text{sigm}()$ DAE: $L = 100$ DAE: $\sigma = 0.01$ DAE: $\eta = 0.005$ SVM: $C = 0.0001$
eNT	LSPC: $\sigma = 0.5$ LSPC: $\lambda = 0.3$	SSA: $P = 50$ SVM: $C = 7.5 \cdot 10^{-5}$	DAE: $f, g = \text{sigm}()$ DAE: $L = 120$ DAE: $\sigma = 0.01$ DAE: $\eta = 0.001$ SVM: $C = 0.005$

(i) *SVM*: a linear SVM is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$. Model selection is performed on a single hyper-parameter: the soft cost C is chosen in the set $\{2.5{\cdot}10^{-5}, 5{\cdot}10^{-5}, 7.5{\cdot}10^{-5}, ..., 0.5, 0.75, 1.0\}$ following Eyben et al. (2016).

(ii) *SF+SVM*: SF is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ using an early stopping criterion. Model selection is performed on a single hyper-parameter: the learned features L is chosen in the set $\{20, 50, 80, 100, 120\}$ in order to explore a coarse-grained grid space around the original number of features $M = 72$. SVM is trained as for system (i).

(iii) *PSF+SVM*: PSF is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}, \mathbf{Y^{tr}}\}$ using an early stopping criterion. Model selection is performed on three hyper-parameters: the non-linearity g chosen in the set $\{\sin(), \cos()\}$, the learned feature L in the set $\{20, 50, 80, 100, 120\}$ as before, the loss parameter λ in the set $\{0.8, 0.9, 0.95, 1.0, 1.05, 1.1, 1.15, 1.2, 1.3, 1.4, 1.6, 1.8,\ 2.0, 2.2, 2.4, 2.6, 2.8\}$ in order to explore a finer-grained grid space around the value $\lambda = 1$. The dimensionality of the learned space is given by the number of features L, evenly divided between the two classes, plus a fixed number of 10 features to account for unlabelled samples. SVM is trained as for system (i).

(iv) *IW+LSPC*: IW is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$ setting the number of basis to the cardinality of the target set $\mathbf{X^{tar}}$. Model selection is performed on two hyper-parameters: the candidate σ chosen in the set $\{0.1, 0.2, 0.5, 1, 2, 3\}$ and the candidate λ in the set $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10\}$ following Hachiya et al. (2012). LSPC is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$. Model selection is performed on two hyper-parameter: the candidate σ chosen in the set $\{0.1, 0.2, 0.5, 1, 2, 3\}$ and the candidate λ in the set $\{0.1, 0.17, 0.32, 0.56, 1\}$ following Hachiya et al. (2012).

(v) *SSA+SVM*: SSA is trained on $\{\mathbf{X^{tr}}, \mathbf{X^{tar}}\}$. Model selection is performed on a single hyper-parameter: the number of PCA components P chosen in the set $\{35, 50, 70\}$ in order to explore a coarse-grained grid below the original number of features $M = 72$. SVM is trained as for system (i).

(vi) *DAE+SVM*: DAE is trained on $\{\mathbf{X^{tr}}, \mathbf{Y^{tr}}\}$ for 10000 epochs and the noise set to a Gaussian with zero mean. Model selection is performed on four hyper-parameters: the non-linearity (for both encoding and decoding) is chosen in the set $\{\text{sigmoid}(), \tanh()\}$ following the standard in the neural network literature, the learned features L in the set $\{70, 100, 120\}$ in order to explore a coarse-grained grid equal or above the original number of features $M = 72$, the variance of the noise σ in the set $\{0.01, 0.05, 0.1\}$ and the learning rate η in the set $\{0.0005, 0.001, 0.005\}$ following the standard in the neural network literature.

Hyper-parameters are selected by cross-validation using a standard grid search method.

Experimental Results

Table 3 reports the hyper-parameter configurations selected by model selection.

References

Eyben, F., Scherer, K. R., Schuller, B. W., Sundberg, J., André, E., Busso, C., … others (2016). The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing. *IEEE Transactions on Affective Computing*, *7*(2), 190–202.

Fernando, B., Habrard, A., Sebban, M., & Tuytelaars, T. (2013). Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the ieee international conference on computer vision* (pp. 2960–2967).

Hachiya, H., Sugiyama, M., & Ueda, N. (2012). Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, *80*, 93–101.

Ngiam, J., Chen, Z., Bhaskar, S. A., Koh, P. W., & Ng, A. Y. (2011). Sparse filtering. In *Advances in neural information processing systems* (pp. 1125–1133).

Schuller, B., Vlasenko, B., Eyben, F., Wollmer, M., Stuhlsatz, A., Wendemuth, A., & Rigoll, G. (2010). Cross-corpus acoustic emotion recognition: Variances and strategies. *IEEE Transactions on Affective Computing*, *1*, 119-131.

Zhang, Z., Deng, J., & Schuller, B. (2013). Co-training succeeds in computational paralinguistics. In *Proceedings of acoustics, speech and signal processing.*