Edge Detection in Noisy Images Using the Support Vector Machines

Javier Redolfi [1]

[1] Centro de Investigación en Informática para la Ingeniería
Universidad Tecnológica Nacional

Curso de Clasificación de Patrones, 2012

Contenido

Contenido

Métodos Clásicos para la Detección de Bordes.

- Se basan en la búsqueda de máximos en las derivadas locales de las imágenes
  - Sobel
  - Prewitt
  - Canny
    - Aplica un prefiltrado para suavizar la imagen y reducir el efecto del ruido
- No son adecuados ante la presencia de ruido impulsivo (ruido conocido como *salt&pepper noise*)

Contenido

Métodos Propuesto para la Detección de Bordes.

- El método esta basado en el trabajo [2]
- No trata de aproximar la derivada como hacen los métodos clásicos
- La idea es entrenar un discriminante lineal para la detección de bordes
- La herramienta utilizada es la Máquina de Soporte Vectorial (SVM)
- Se utilizan dos clases de SVM
  - La primera se encarga de la reducción del ruido (Regresión)
  - La segunda realiza la detección de bordes (Clasificación)
- Se obtienen resultados similares a los métodos previos en imágenes sin ruido
- Se obtienen resultados superiores en imágenes ruidosas

Contenido

SVM para clasificación - SVC

- Encontrar una función $y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ que cumpla con
  - $y(\mathbf{x}_n) > 0 \quad \forall \quad t_n = +1$
  - $y(\mathbf{x}_n) < 0 \quad \forall \quad t_n = -1$
  - Donde x_n son los vectores de entrenamiento con sus correspondientes etiquetas t_n, para $n = 1, \ldots, N$
  - $\mathbf{x}_n \in \mathbb{R}^M, \quad t_n \in \{-1, +1\}$
  - Notar que: $y(\mathbf{x}_n) t_n > 0 \quad \forall \quad n$
- Esta formulación supone que los datos son linealmente separables en alguna transformación de nuestro espacio de características

SVM para clasificación - SVC

- La distancia del punto más cercano al hiperplano que separa las clases es

$$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \frac{t_n \left[\mathbf{w}^T \phi(\mathbf{x}_n) + b\right]}{\|\mathbf{w}\|} = r$$

- Escalando $\mathbf{w}$ y b adecuadamente podemos obtener

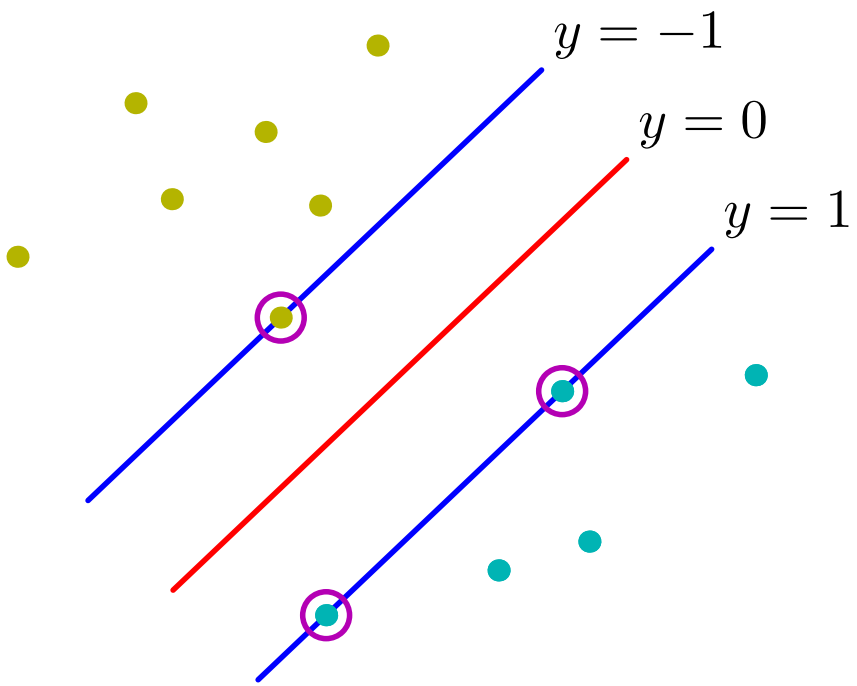
$$t_n \left[\mathbf{w}^T \phi(\mathbf{x}_n) + b\right] = 1$$

- Para los que están más lejanos tenemos

$$t_n \left[\mathbf{w}^T \phi(\mathbf{x}_n) + b\right] \geq 1$$

- Imagen

SVM para clasificación - SVC



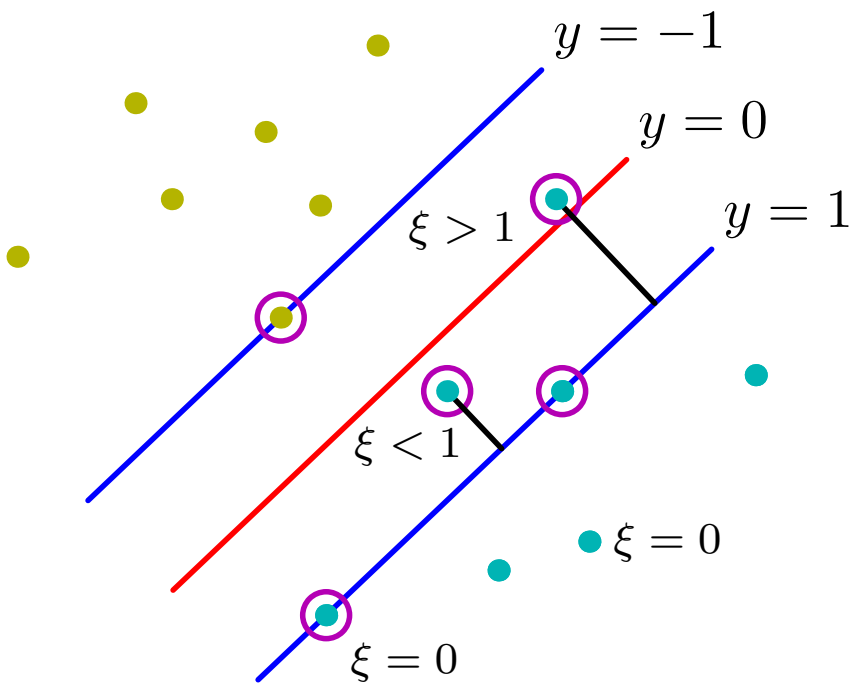
SVM para clasificación - SVC - No separable

- Cuando el problema no es linealmente separable
- Definimos las *slack* variables

$$\xi_n = \begin{cases} 0 & \text{si } y(\mathbf{x}_n) \text{ está en el lado correcto} \\ |t_n - y(\mathbf{x}_n)| & \text{en otro caso} \end{cases}$$

- Imagen

SVM para clasificación - SVC



SVM para clasificación - SVC - Optimización

- Debemos optimizar

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2} \|w\|^2$$

- Sujeto a las restricciones

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n$$

- Para $C \to \infty$, $\xi_n = 0$ y recuperamos el problema planteado anteriormente

SVM para clasificación - ν-SVC

- Esta formulación de SVM es conocida como C-SVC
- ¿Cómo interpretamos C?
- Existe otra formulación del problema conocida como ν-SVC
  - Tenemos un nuevo parámetro ν, el cual reemplaza a C
  - ν representa la fracción de puntos que permitimos que estén del lado incorrecto

Contenido

SVM para regresión - SVR

- Las SVM también puede ser utilizadas para resolver el problema de regresión
- Esto es, obtener una función que optimice

$$\frac{1}{2}\sum_{n=1}^{N}(y_n - t_n)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

- Para obtener una solución *sparse*, reemplazamos la función de error cuadrática por la función ε-*insensitive*

SVM para regresión - SVR - ε-*insensitive*

- Función ε-*insensitive*

$$E_\varepsilon(y(\mathbf{x}) - t) = \begin{cases} 0 & \text{si } |y(\mathbf{x}) - t| < \varepsilon \\ |y(\mathbf{x}) - t| - \varepsilon & \text{en otro caso} \end{cases}$$

- La función de error ε-*insensitive* es insensible hasta que el error es mayor en valor absoluto que ε
- Planteado el nuevo problema de optimización a resolver

$$C \sum_{n=1}^{N} E_\varepsilon(y(\mathbf{x}_n) - t_n) + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

SVM para regresión - SVR - No separable

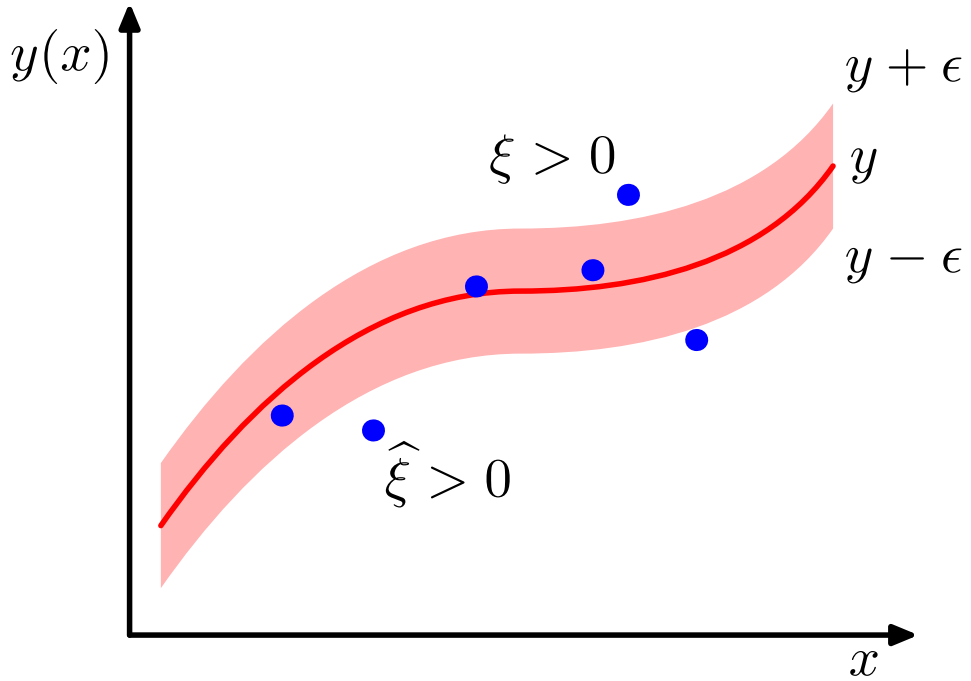
- Igual que en SVM para clasificación podemos plantear una solución "floja" introduciendo las *slack* variables
- Para el caso de regresión necesitamos 2 variables, porque cometemos error cuando estamos muy por encima o muy por debajo del hiperplano

$$\xi_n > 0 \quad \text{y} \quad \hat{\xi}_n = 0, \quad \text{cuando} \quad t_n > y(\mathbf{x}_n) + \varepsilon$$

$$\hat{\xi}_n > 0 \quad \text{y} \quad \xi_n = 0, \quad \text{cuando} \quad t_n < y(\mathbf{x}_n) - \varepsilon$$

- Imagen

SVM para regresión - SVR



SVM para regresión - SVR - Optimización

- El problema de optimización es ahora

$$C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) + \frac{1}{2}\|\mathbf{w}\|^2$$

- Sujeto a las restricciones

$$\xi_n \geq 0, \quad \hat{\xi}_n \geq 0$$
$$t_n \leq y(\mathbf{x}_n) + \varepsilon + \xi_n$$
$$t_n \geq y(\mathbf{x}_n) - \varepsilon - \hat{\xi}_n$$

SVM para regresión - SVR

- El valor de ε me define un tubo de insensibilidad en la función de error
- Ver figura anterior
- Este tipo de máquina lineal se conoce como ε-SVR
- Igual que en el caso de SVC, existe una formulación alternativa conocida como ν-SVR

Contenido

Radial Basis Function - RBF

- Las RBF son una familia de funciones que depende solo de la distancia del punto a un centro definido
- Típicamente la distancia utilizada es la Euclidiana

$$\phi_j(\mathbf{x}) = h(\|\mathbf{x} - \mu_j\|)$$

- Una RBF comúnmente usada como kernel en SVM es

$$k(\mathbf{x}, \mathbf{x}') = e^{-\gamma|\mathbf{x}-\mathbf{x}'|}$$

Contenido

Introducción a libsvm

- libsvm es una librería escrita en C++ para trabajar con SVM
- Permite
  - Clasificación Binaria basada en Vectores de Soporte - SVC
  - Regresión basada en Vectores de Soporte - SVR
  - Estimación de Distribuciones - one-class SVM
  - Clasificación Multiclase basada en Vectores de Soporte - SVC
- `http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html`
- Tiene interfaz con los lenguajes de programación más comunes
- paralelización rápida con openmp

Introducción a libsvm - Herramientas

- Herramientas Principales
  - svm-train
    - Utilizada para resolver el problema de optimización dada la muestra en entrenamiento, obtención del modelo
  - svm-predict
    - Utilizada para realizar predicciones de las muestras de test usando el modelo ya entrenado

Contenido

Reducción de Ruido

- La idea es reemplazar los pixeles de la imagen ruidosa por un nuevo valor
- El nuevo valor se obtiene utilizando regresión SVM
- Para entrenar el modelo suponemos que el valor de un pixel está relacionado con los valores de sus 8 vecinos
- Primero generamos una imagen en escala de grises
- Luego le agregamos ruido en posiciones conocidas
- Por último entrenamos la SVM utilizando ε-SVR y un kernel RBF

Reducción de Ruido - Vectores de Entrenamiento

p_1	p_2	p_3
p_4	p_5	p_6
p_7	p_8	p_9

Figura: Imagen Sin Ruido

n_1	n_2	n_3
n_4	n_5	n_6
n_7	n_8	n_9

Figura: Imagen Ruidosa

- Vector de Entrenamiento

$$\mathbf{x}_n = [n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9]$$

- Valor Verdadero

$$t_n = p_5$$

- Ejemplo

$$\mathbf{x}_n = [10, 11, 9, 11, 0, 10, 11, 11, 9]$$

$$t_n = 9$$

Reducción de Ruido - Imágenes de Entrenamiento

- Imágenes Usadas para el Entrenamiento

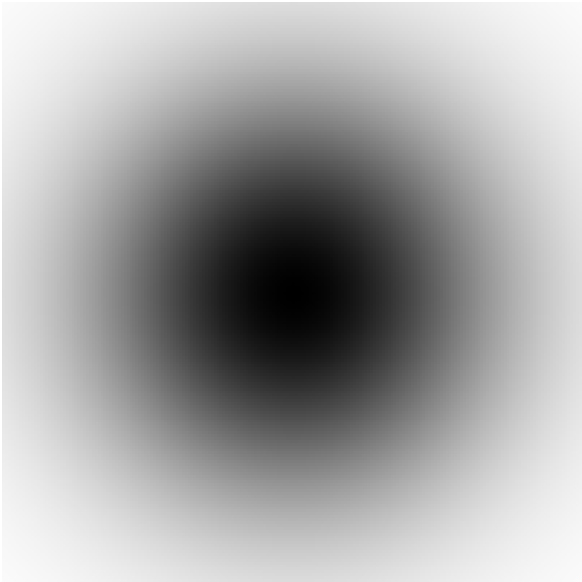


Figura: Valores Verdaderos

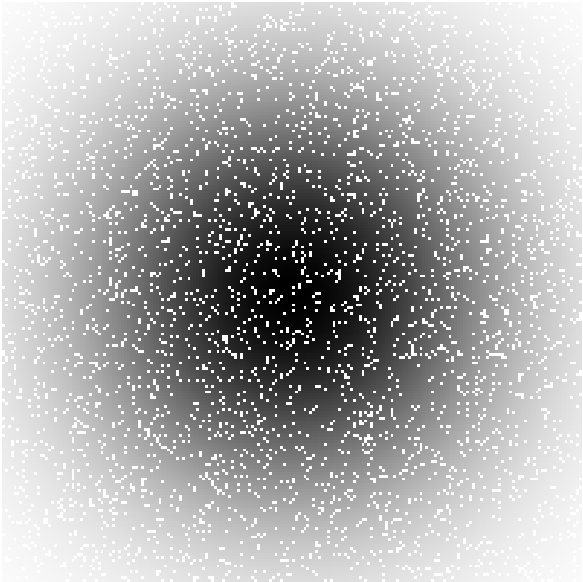


Figura: Vectores de Train

Reducción de Ruido - Resultados

Figura: Imagen Original

Figura: Imagen Ruidosa

Figura: Resultado usando ε-SVR

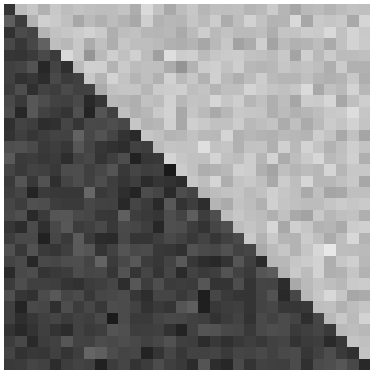
- Parámetros de Entrenamiento: $C = 1, \gamma = 10$

Contenido

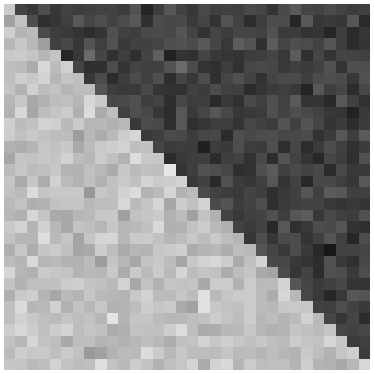
Detección de Bordes

- Entrenamos un SVC para clasificar cada pixel de una imagen como borde o no borde
- El nuevo valor se obtiene utilizando clasificación SVM
- Suponemos que la condición de borde de un pixel está relacionada con el valor del pixel y los valores de sus 8 vecinos
- Pero lo que nos importa es la diferencia en los valores de los pixeles
- Primero generamos 4 imágenes con los bordes ubicados en posiciones conocidas
- Entrenamos la SVM utilizando C-SVC

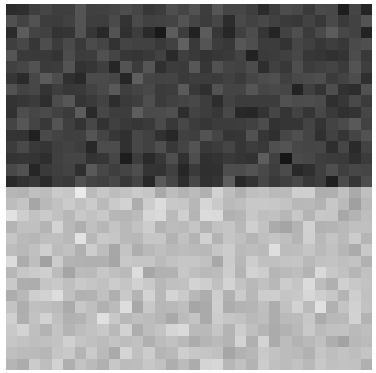
Detección de Bordes - Vectores de Entrenamiento



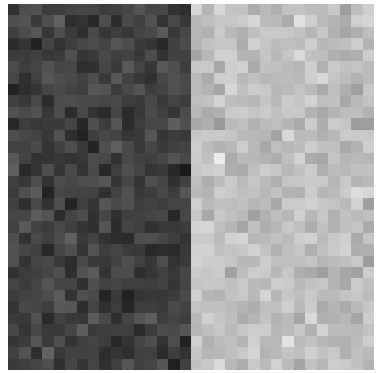
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



0	1	0	0
0	0	1	0
0	0	0	1
0	0	0	0



0	0	0	0
1	1	1	1
0	0	0	0
0	0	0	0



0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0

Detección de Bordes - Vectores de Entrenamiento

- Vector de Entrenamiento

$$\mathbf{x}_n = [n_1 - n_5, n_2 - n_5, n_3 - n_5, n_4 - n_5, n_6 - n_5, n_7 - n_5, n_8 - n_5, n_9 - n_5]$$

- Valor Verdadero

$$t_n \in \{0, 1\}$$

- Ejemplo de vector borde

$$\mathbf{x}_n = [-0{,}01, 0{,}47, 0{,}46, -0{,}00, 0{,}50, -0{,}06, -0{,}02, -0{,}05]$$

$$t_n = 1$$

Detección de Bordes - Resultados

Figura: Imagen Original

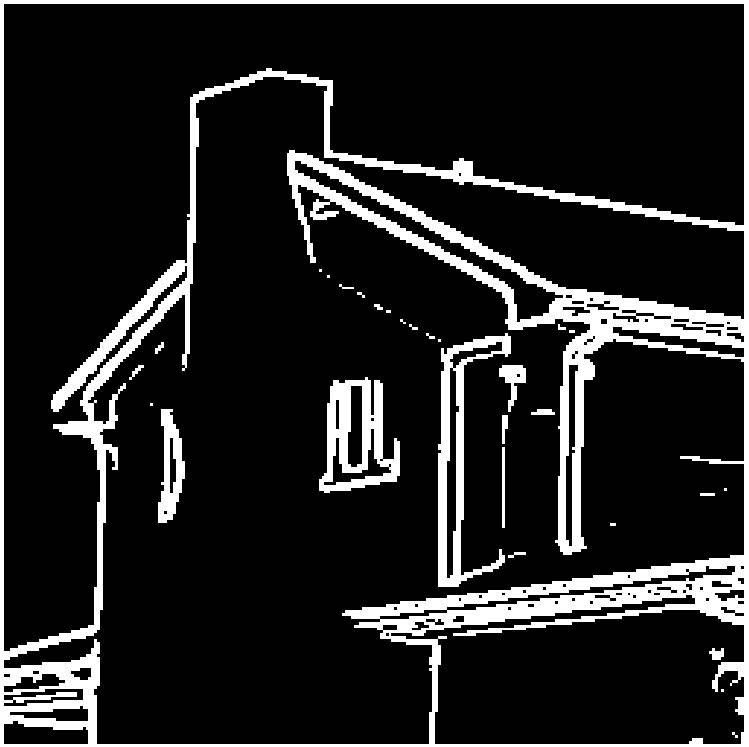
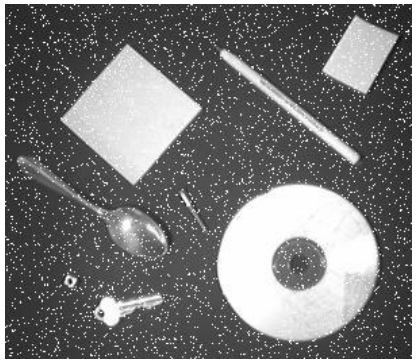


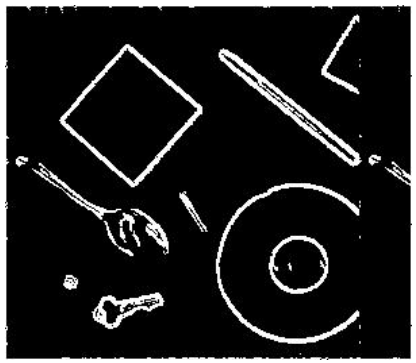
Figura: Imagen de Bordes

- Parámetros de Entrenamiento: $C = 1000$, $\gamma = 1$

Resultados





Resultados - Comparación con los Métodos Clásicos

Figura: SVM

Figura: Sobel

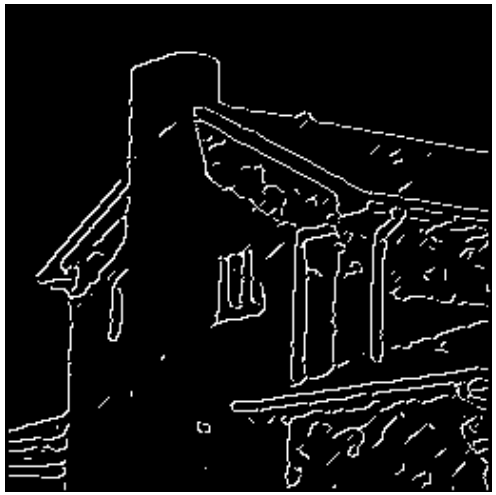


Figura: Canny

Conclusiones

- ¿Casos reales de este tipo de ruidos?
- Demasiados parámetros para ajustar
  - Imagen para reducción de ruido
  - Imagen para entrenamiento de borde
  - SVC
  - SVR

C. M. Bishop.
Pattern recognition and machine learning.
Springer, 1st ed. 2006. corr. 2nd printing edition, Oct. 2006.

H. Gómez-Moreno, S. Maldonado-Bascón, and F. López-Ferreras.
Edge detection in noisy images using the support vector machines.
In *Proceedings of the 6th International Work-Conference on Artificial and Natural Neural Networks: Connectionist Models of Neurons, Learning Processes and Artificial Intelligence-Part I*, IWANN '01, pages 685–692, London, UK, UK, 2001. Springer-Verlag.