Donde x, es la señal de entrada, N su longitud y $\hat{r}_+[x]$ y $\hat{r}_-[x]$ los retardos positivos y negativos de la función de correlación respectivamente.

B. Simetría de la función de autocorrelación

Dadas las características particulares de simetría en la función de autocorrelación, para cada clase observada en un análisis preliminar de las señales, se calcula la medida de simetría de la correlación con la formulación presentada en el apéndice A.

C. Función de autocorrelación en frecuencia

Nuevamente, tomando las señales producidas por el análisis ondita, primero se pasan al dominio de la frecuencia mediante FFT y, luego del mismo modo que antes, se calcula la función de autocorrelación en frecuencia.

D. Momentos

A las 6 medidas descritas anteriormente, se les extraen momentos estadísticos de orden superior, con el fin de alcanzar una mejor capacidad de discriminación entre ellas para las diferentes clases. El análisis de los momentos de las 6 medidas mostró que el momento 39 y 40 genera lo que se considera una buena agrupación y separación de las clases.
Con las 3 medidas descritas anteriormente y teniendo en cuenta que se extraen tanto a los coeficientes de aproximación como a los de detalle, se obtienen 6 vectores a los que se les calculan los momentos 39 y 40 respectivamente. Esto conforma un espacio de características de dimensiones (rasgos)x(observaciones), es decir 12x48963, del cual, se sustrajeron 4000 observaciones al azar con el fin de entrenar los clasificadores implementados.

V. DESARROLLO DE LOS CLASIFICADORES

A. Preliminares

Es de anotar, que en todo el espacio de características obtenido se contienen números muy pequeños, puesto que todas las funciones de autocorrelación fueron normalizadas, mientras los momentos son de órdenes relativamente altos (39 y 40). La normalización es necesaria, por cuanto un mismo evento sísmico no se presenta, de igual manera, en todas las estaciones. Por esto último, se plantea un mapeo no lineal de la siguiente forma:

$$\phi_m = \log_{10}(abs(\phi))$$

Con ϕ_m y ϕ el espacio mapeado y original respectivamente.

La operación anterior garantiza que el espacio de características tenga valores pequeños y, dentro de un rango reducido, además evita errores de propagación en las operaciones de punto flotante, ya que la relación de variaciones tan pequeñas pueden resultar en que los valores no tengan efecto o se vuelvan cero. El valor absoluto asegura que el logaritmo no tenga valores negativos

tipo	ava	ice	lp	re	tl	tre	ts	vt
ϕ	171	1410	1542	1302	763	8	114	1378

Tabla 1

Otro factor importante a tener en cuenta, antes de desarrollar los clasificadores, consiste en conocer exactamente dentro del espacio de las características, cuantas de las muestras pertenecen a cada clase. Como se observa en la tabla 1, existe un claro desbalance en la proporción de todas las clases. Así, para el tipo tremor, solo se tienen 8 en entrenamiento y 89 en validación, por lo tanto,

no se va a clasificar, por lo menos con SVM. Debido precisamente al desbalance de las clases, la medida del error en los clasificadores se extrae de la forma mostrada en el apéndice B.

B. Árboles de clasificación (CART)

Como representante de los clasificadores estadísticos se realizó una implementación en fortran 95 para un Cluster SPARC Solaris del algoritmo para un árbol de clasificación basados en [1] y [8] con: $\max n = 2$ y se obtuvieron los siguientes resultados: 1087 nodos para el sub-árbol 10 con alfa 4.85e-12.

clase	% err clase	% err total
ava	70.67	1.2254
ice	72.41	7.1483
lp	79.92	15.667
re	42.6	13.249
tl	52.31	8.6744
tre	100	0.1817
ts	54.72	1.443 9
vt	23.25	4.2542

Tabla 2

El programa solo fue desarrollado para este problema en Solaris debido a los altos requerimientos de memoria. Según los reportes obtenidos, 2.4Gbytes de memoria física y 2.1Gbytes de memoria virtual. En pruebas preliminares con espacios de características pequeños, la ganancia de tiempo de ejecución de Solaris respecto a Windows es de mas o menos 30%.

C. Redes neuronales artificiales (RNA)

Como representante de los clasificadores llamados de inteligencia artificial se realizó una implementación en fortran 95 de algoritmo para plataforma PC Windows, Cluster SPARC Solaris y se obtuvieron los siguientes resultados. La red neuronal implementada tiene una sola salida, por lo tanto se implementó una de ellas para cada clase con las siguientes características: 12 entradas, 16 nodos ocultos, $\mu = 1e-3$, $\beta = 10$ y 200 épocas.

clase	% err clase	% err total
ava	44.2	1.966
ice	48.6	3.242
lp	40.4	4.492
re	37.2	7.446
tl	39.8	9.294
tre	----	-----
ts	43.0	5.738
vt	45.6	6.592

Tabla 3

La implementación se hizo conforme a [10], [11] y [12] con Levenberg-Marquardt como algoritmo de entrenamiento y Nguyen-Widrow para inicialización de pesos.
La ganancia en velocidad de ejecución de la versión Solaris respecto a la Windows es al rededor de 30-40%.

D. Máquinas de soporte vectorial (SVM)

Este es el clasificador central del trabajo con una implementación en fortran 95 del algoritmo de máquinas de soporte vectorial. Como la SVM esta propuesta para una única salida se hicieron realizaciones independientes para cada clase. Los valores iniciales para ó y C se tomaron conforme a [16]. los resultados obtenidos son los siguientes:

clase	sigma	C	SV	% err clase	% err total
ava	3.0d0	1.5d0	1308	8.309	2.128
ice	2.5d0	1.5d2	1530	6.254	1.261
lp	1.0d1	1.0d0	1238	6.078	5.629
re	7.0d1	1.5d1	1752	3.666	5.902
tl	1.0d1	2.0d2	1053	6.825	4.781
tre	-----	-----	----	-----	-----
ts	1.3d3	2.0d2	1890	11.23	2.691
vt	1.6d2	2.0d2	1955	7.070	4.096

Tabla 4

El algoritmo fue implementado solo para Windows debido a que los tiempos de ejecución en el peor de los casos (con los parámetros de la SVM mal condicionados) no superan los 15 minutos y en condiciones normales son entre 2 y 5 minutos. Es mas a veces tarda mas evaluando el grupo de validación ya que son bastantes y en algunos el número de vectores de soporte es alto. La máquina de soporte vectorial se hizo conforme a [13], [14] y [15] con un algoritmo de minimización utilizando NPA (Nearest Point Algorithm), tratando el problema de minimización cuadrática como uno de punto cercano y función Kernel RBF conforme a [18]. En contraste a NPA también se hicieron algunas pruebas con SMO (ver [17]) con resultados muy similares pero con tiempos de proceso sustancialmente mayores.

Para la regla de búsqueda de los mejores parámetros ó y C no se utiliza el porcentaje de error de clasificaciones incorrectas sino la siguiente formulación:

$$e_{\%b} = 100\left(\frac{N_{m1}}{2N_1} + \frac{N_{m2}}{2N_2}\right)$$

Donde N_{m1} y N_{m2} número de clasificaciones erróneas de la clase 1 y 2 respectivamente, N_1 y N_2 número total de datos por clase a clasificar (e.g. si se va a clasificar el tipo ava, la clase 1 son todos los ava y la dos todos los demás) y %eC porcentaje de error de clasificación. Obsérvese que en la ecuación anterior cada clase aporta un 50% de error sin importar el número de datos en cada clase, así de este modo garantizamos que el error se minimice siempre y cuando las clasificaciones erróneas estén balanceadas.

VI. CONCLUSIONES

El desempeño de los diferentes tipos de clasificadores implementados (árboles de clasificación, redes neuronales y SVM), en la clasificación de eventos sísmicos mostró un comportamiento notablemente superior para las máquinas de soporte vectorial. Los problemas de clasificadores, en términos generales, requieren efectiva solución a tres aspectos: garantizar un alto grado de confiabilidad en cuanto a la capacidad de generalización, el error en el clasificador debe depender no solo del error cuadrático medio MSE con la finalidad de obtener resultados equilibrados, el tiempo de ejecución para la convergencia del algoritmo y la complejidad computacional del mismo.

Los argumentos anteriormente mencionados, junto al procedimiento de optimización (NPA) sobre la función de costo no depende de efectos aleatorios que puedan hacer que el algoritmo no converja o permanezca indefinidamente en un mínimo local y que la estructura de la máquina de soporte vectorial no la especifica el usuario, el número de pesos (multiplicadores lagrangianos) se