Tarea 4

13 de mayo de 2019

Alejandro Quiñones

1. Motivación

La tarea sirve como una introducción a redes neuronales, una de las herramientas más importantes y populares actualmente en el área de la inteligencia artificial. Se puede apreciar la importancia de encontrar buenos hiperparámetros, y la complejidad y carga computacional que eso conlleva. Por el otro lado, podemos comparar tanto los resultados como la ejecución de una red neuronal básica, MLP, con otro clasificador, SVM.

2. Solución propuesta

Con respecto a la implementación de la red neuronal, se usó un simple MLP con tres capas ocultas. Es posible modificar ciertos hiperparámetros y partes del modelo que afectan tanto el proceso de aprendizaje como la optimización lograda: La cantidad de neuronas en cada capa oculta, la función de activación después de cada capa, el algoritmo de optimización usado, el término de regularización (α) y si el learning rate será variado o constante.

Por otro lado, para los experimentos realizados sobre el SVM podemos cambiar el kernel, modificar el coeficiente del kernel (γ) y el castigo al error (C), y por último si tenemos el kernel *poly* podemos especificar el *degree* de su polinomio.

3. Experimentos realizados

Para el MLP, se consideraron los siguientes casos:

1. Cantidad de neuronas: $(50, 50, 50)$, $(50, 100, 50)$, $(8, 8, 8)$ y $(8, 10, 8)$
2. Función de activación: *tanh* y *ReLU*
3. α: 0,0001 y 0,05
4. Algoritmo de optimiazción: *SGD* y *Adam*
5. Learning rates: *Constant* y *Adaptive*

Para SVM, se realizaron las siguientes variaciones:

1. Kernel: rbf, *Sigmoid* y *poly*
2. γ: 2^{-8}, 2^{-6}, 2^{-4},2^{-2} y *auto*
3. C: 2^{-2}, 2^{0}, 2^{2} y 2^{4}
4. *degree*: 0, 1, 2, 3

Para SVM, el modelo final fue el con las variables que entregaban mayor *accuracy* en el set de validación, después de haber entrenado en el set de entrenamiento. Específicamente, se utilizó un kernel rbf con un $\gamma = 2^{-6}$y $C = 2^2$.

Para MLP, se mezclaron los sets de entrenamiento y validación y luego se dividieron nuevamente en la misma proporción. Esto se hizo debido a que permitió que la implementación se haga bajo ciertas librerías que permitían la paralelización del código. Las variables resultantes fueron una configuración de 50, 100 y 50 neuronas por capa, un $\alpha = 0{,}0001$, una función de activación *tanh* y un learning rate adaptativo, resuelto bajo *Adam*. Los resultados se muestran en la Tabla 1.

Modelo	Accuracy en validación	Accuracy en test
MLP	88.41 %	84 %
SVM	89.5 %	87.33 %

Tabla1. Resultados finales de los experimentos en SVM y MLP.

4. Conclusión

Como conclusión, podemos confirmar la complejidad de encontrar buenos valores para los hiperparámetros. La búsqueda significa amplificar el tiempo de ejecución por la cantidad de combinaciones posibles, pero hacen el modelo mucho más robusto. Por otro lado, podemos notar que los resultados de SVM son similares (incluso mejores) a los de MLP. Esto permite ejemplificar el Teorema de "No Free Lunch": MLP no es universalmente mejor que SVM aunque en la actualidad se presente con mayor adopción. Existen casos donde modelos más complejos no resuelven de mejor manera el problema, especialmente cuando se varía el tamaño de los datasets.