ISSN 1870-4069

Evaluación de candidatos para la retroalimentación de corpus por medio de bagging

Cecilia Reyes Peña, David Pinto Avendaño, Darnes Vilariño Ayala

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

reyesp.cecilia@gmail.com, {dpinto,darnes}@cs.buap.mx

Resumen. Las redes sociales contienen suficiente elementos para construir corpus acerca de temas novedosos,en dichos corpus existe la posibilidad de que se conviertan en obsoletos debido a la naturaleza efímera de dicha información. La tarea de retroalimentar corpus para mantenerlos vigentes es muy importante y a su vez, una tarea muy difícil cuando se hace manualmente debido a la cantidad de información que se tiene que manejar. En este trabajo se presenta la comparativa de los resultados de la aplicación de la técnica de ensamble de clasificadores Bagging con votación simple para la selección de Tweets candidatos con la finalidad de retroalimentar el corpus de entrenamiento. Dicho corpus está balanceado y dividido en cuatro clases (alegría, tristeza, ira y miedo) y el proceso de clasificación es realizado por medio de tres modelos: Ranking, Naïve Bayes y Probabilidad de Bigramas. La candidatos son seleccionados de un conjunto de prueba etiquetado manualmente y la retroalimentación del corpus será evaluada por medio de pruebas K-Fold Cross Validation.

Palabras clave: bagging, ranking, Naïve Bayes, probabilidad de bigramas, retroalimentación de corpus, ensamble de clasificadores.

Candidates Evaluation for Corpus Feedback by Bagging

Abstract. The social networks contains enough elements for build corpus about novel topics in which there is the possibility of become obsolete because to their fleeting nature. The feedback corpus task for keeping them current is very important and in turn a difficult and cost task because the used information amount for this. In this work, the comparative among the results of classifier ensemble technique with simple voting in order to candidates Tweets selection for train corpus feedback has been presented. The train corpus is divided in four classes (happiness, sadness, anger and fear) and the classification process is performed by three models: Ranking, Naïve Bayes and Bigrams Probabilities. The

candidates are selected from a manually tagged test set and the feedback is evaluated by K-Fold Cross Validation.

Keywords: bagging, ranking, Naïve Bayes, bigrams probability, corpus feedback, classifier ensemble.

1. Introducción

Las redes sociales como Twitter contienen una gran cantidad de información cambiante, es decir, los temas manejados dentro de estas son recientes y solo se mantienen vigentes por cortos lapsos de tiempo. Al trabajar con corpus de dicha información es necesario mantenerlos actualizados, lo cual es un proceso muy dificil y costoso si se hace manualmente. Existen técnicas como Bootstrapping para retroalimentar corpus de forma dinámica, por medio de un proceso de clasificación de información a base de etiquetado partiendo de lo particular a lo general, es decir parte de un elemento semilla para extraer un conjunto mayor de información que será clasificado mediante uno o varios modelos de clasificación [1]. Para que esto sea posible es necesario implementar estrategias de selección de elementos candidatos por medio de un clasificador, el cual considera algunas características y desprecia otras. Una forma de aprovechar múltiples características de la información es a través de la implementación de múltiples clasificadores que puedan aportar resultados según su naturaleza, siendo esta una de las principales ventajas del ensamble de clasificadores.

Se puede definir al ensamble de clasificadores como un trabajo colaborativo entre diversos modelos de clasificación dentro de una misma tarea, de tal forma que dicha colaboración ofrezca mejores resultados respecto a los que puedan otorgar cada uno de los clasificadores participantes por separado, simulando la naturaleza humana de pedir opiniones [2]. Existen diferentes técnicas de ensamble y entre las más populares podemos mencionar al Bagging, en el cual se generan múltiples versiones de una predicción de clase, la predicción final será definida mediante un proceso de votación. En términos generales, las diferentes versión se crean a partir de la agregación de elementos tomados al azar del conjunto de entrenamiento para generar nuevos conjuntos de entrenamiento que varían en cada iteración obteniendo así las versiones de la predicción [3]. Otra técnica popular es el boosting en el cual se le asigna peso a cada ejemplo y en cada iteración se modifica, al finalizar las iteraciones se realiza una votación de los resultados considerando los pesos finales [10].

En este trabajo se compara los resultados de la técnica bagging aplicada a un proceso de selección de tweets candidatos para la retralimentación de un corpus etiquetado manualmente en cuatro clases (alegría, tristeza, ira y miedo), mediante tres modelos de clasificación: uno basado en la técnica de recuperación de la información Ranking, Naïve Bayes y Probabilidad de bigramas.

Este documento está dividido en las siguientes secciones: la sección 2 contiene los trabajos relacionados al ensamble de clasificadores, la sección 3 describe los clasificadores utilizados en este trabajo, la sección 4 contiene los resultados de

la evaluación de los candidatos proporcionados por el ensamble, por último la sección 5 contiene las conclusiones de este trabajo.

2. Trabajos relacionados

Dentro del campo del procesamiento del lenguaje natural, la técnica de ensamble de clasificadores ha sido utilizada en diversos proyectos, uno de ellos es la mejora de procesos de análisis de opinión afectiva, mediante elementos morfosintácticos para la búsqueda de relaciones avanzadas difíciles de detectar, utilizando algoritmos de aproximación simple, basado en DAL (Diccionario de Afecto en el Lenguaje), basado en otro diccionario de mayores dimensiones y un árbol de sintáxis con información morfológica. Para la clasificacion se utiliza el algoritmo MaxEnt (Máxima Entropía) y para el ensamble se utiliza la técnica bagging con muestras del 80 % para entrenar y los resultados son analizados en un nuevo clasificador [8]. Otra aplicación del ensamble es durante la extracción de características de rostro y audio por medio de los algoritmos PCA-engenfaces y MFCC respectivamente, los clasificadores utilizados fueron Redes Bayesianas, Naïve Bayes, K-vecinos cernos, Redes Neuronales y Árboles de desición; para el ensamble entre ellos se utilizó la técnica Staking, en la cual los clasificadores tienen una jerarquía y la predicción de los clasificadores mas altos influye en los mas bajos [9]. También se ha aplicado el ensamble en la identifiación de correos electrónicos denomidaods spam, donde los datos utilizados fueron extraídos de corpus públicos considerando de estos solo el contenido que el usuario puede leer representado en vectores de relevancia, para la clasificación se utilizaron los modelos: SVM (Support Vector Machine), Árbol de desición, Red Neuronal, K vecinos cercanos, Naïve Bayes, entre otros. El ensamble se realizó mediante la técnica Boosting apoyando la desición por medio de votos simples [11]. GuoDong Zhou et al. [12] presenta un modelo de reconocimiento de nombres de genes y protenias en texto considerando también sus abreviaturas por medio de un módulo donde cada término sea empatado con un diccionario. Los modelos de clasificación utilizados son: SVM y dos modelos ocultos de Markov discriminatorios, dejando la desición final a una votación mayoritaria simple. Por su parte, Onan Aytuğ et al. [13] presenta la comparativa de varias técnicas de ensamble como AdaBoost, Bagging, Dagging, Random Subspace y Votación mayoritaria simple; en la tarea de extración de palabras clave usando los modelos de clasificación Naïve Bayes, SVM, Regresión Logística y Random Forest. Los mejores resultados de esta comparativa fueron los otorgados por la técnica Bagging aplicada al clasificador Random Forest.

3. Clasificadores

En esta sección se describen los clasificadores utilizados para la técnica del ensamble en este trabajo.

3.1. Ranking

El modelo de clasificación utilizado en este trabajo está basado en la técnica de recuperación de la información Ranking simple, donde se aplican algoritmos para crear listas de relevancia basándose en las características propias de una colección de documentos [4]. Para esto se crearon cuatro documentos (uno por clase) para formar un corpus o colección, posteriormente es necesario conocer la frecuencia de cada palabra dentro de cada documento ($TF_{t,d}$) y la relevancia que tiene la misma dentro del corpus (IDF_t, Inverse Document Frequency of the Term), para esta última se emplea la Eq.1, donde N es el total de documentos del corpus, DF_t (Document Frequency of the Term) el número de documentos en el que aparece la palabra y se le aplica el logaritmo base diez para suavizar la relevancia. Para obtener el valor de relevancia final se utiliza la ecuación 2:

$$IDF_t = \log_{10}(N/DF_t), \tag{1}$$

$$TF - IDF_{t,d} = TF_{t,d} \times IDF_t. \tag{2}$$

El proceso para clasificar un tweet nuevo es similar a realizar una consulta, primero se hace un pesado de cada una de las palabras que lo conforman considerando la frecuencia de la palabra en el tweet $TF_{t,q}$ (Term Frequency in the Query) y en el IDF_t de cada documento (Eq. 3), obteniendo así un vector compuesto por el $TF-IDF$ de cada palabra. Finalmente, se realiza un producto punto del vector y el documento que aporte mayor puntaje es al que corresponde el tweet:

$$w_{t,q} = \log_{10}(1 + TF_{t,q} \times IDF_t). \tag{3}$$

Una de las modificaciones del algoritmo de Ranking propuesto en este trabajo es el cálculo de la relevancia de bigramas en un documento. Desde la lectura de los documentos del corpus de entrenamiento, el modelo agrupa bigramas de palabras consecutivas y los toma en cuenta como una sola palabra, al igual que en los tweets a clasificar, el modelo creará los bigramas del tweet y serán evaluados respecto a los almacenados (ver Figura 1).

3.2. Naïve Bayes

Naïve Bayes es uno de los clasificadores estadísticos supervisados más populares. Está basado en el Teorema de Bayes y asume que una característica de una clase en particular contribuye de forma independiente a la probabilidad de que cualquier otra característica pertenezca o no a la misma clase [5]. El funcionamiento del clasificador radica en calcular la probabilidad de pertenencia de cada característica X en cada una de las clases C, cuando se encuentra el valor más alto se retorna el nombre de la clase al cual pertenece el objeto (Ec. 4):

$$\widehat{y} = argmax_{k \in \{1,\ldots,K\}} P(C) \prod_{i=1}^{n} P(x_i \mid C_K). \tag{4}$$

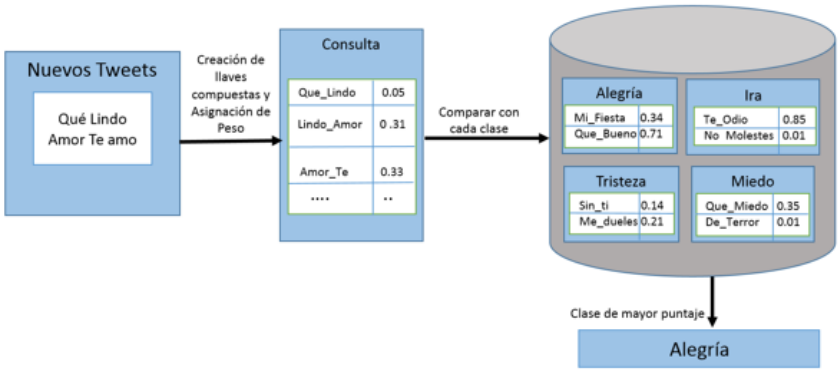

Fig. 1. Modelo Ranking con llaves compuestas.

3.3. Probabilidad de bigramas

Se puede definir al N-grama como una serie de N palabras consecutivas que conforman a una sentencia u oración y se denominan según su grado (el tamaño de la N). En el procesamiento del lenguaje natural, los modelos más usados basados en N-gramas, son comúnmente unigramas (de una palabra), bigramas (de dos palabras) y trigramas (de tres palabras) [6]. Cabe resaltar que entre más grande sea el N-gramas contiene más información que el grado N-1. Para el cálculo de probabilidades en modelos basado en N-gramas, es frecuentemente utilizada la suposición de Markov, en la cual se asume que la probabilidad de una palabra depende solamente de las N-1 palabras anteriores [7]. Por lo tanto, en un modelo de bigramas, la probabilidad de una sentencia ($P(s)$) conformada por N palabras $(w_1, w_2, ..., w_n)$ está dada por la multiplicación de las probabilidades de cada palabra dada la anterior (Ec. 5):

$$P(s) = P(w_1)P(w_2\ |w_1)...P(w_n\ |w_(n-1)). \tag{5}$$

4. Resultados

Para la realización de las prueba se utilizó un corpus de entrenamiento conformado por 105,598 tweets repartidos en forma balanceada en cuatro clases, al evaluarlo mediante k-Fold Cross Validation con una K=10, dando como resultado un porcentaje promedio de aciertos del 90 %. Por su parte, el conjunto de prueba está conformado por 234 tweets (127 de alegría, 31 de tristeza, 62 de ira y 14 de miedo), ambos etiquetados manualmente. Inicialmente se midió el rendimiento de cada clasificador respecto a aciertos y errores en la clasificación por medio de pruebas Presicion, Recall y F-Measure (ver Tabla 1).

En la selección de candidatos del conjunto de prueba, utilizando el corpus de entrenamiento original en los tres modelos de clasificación se obtuvieron los siguientes resultados (ver Tabla 2).

Se realizaron cinco pruebas utilizando el conjunto de prueba y entrenando con Bagging, tomando al azar muestras equivalentes al 80 % del tamaño del corpus de entrenamiento original para cada uno de los tres modelos. En la Tabla 3 se muestra el promedio de candidatos obtenidos para cada clase. Una vez retroalimentado el corpus de entrenamiento, se aplicó la prueba K-Fold Cross Validation para medir la integridad del corpus con un k=5 (ver Tabla 4).

Tabla 1. Resultado de Medidas Presicion, Recall y F-Measure de los tres modelos de clasificación.

Model	Measure	Alegría	Tristeza	Ira	Miedo
Ranking	Precision	0.733	0.4	0.71	0.174
	Recall	0.693	0.387	0.435	0.571
	F-measure	0.712	0.393	0.53	0.266
Naïve Bayes	Precision	0.95	0.195	0.586	0.163
	Recall	0.299	0.548	0.435	0.714
	F-measure	0.455	0.288	0.5	0.266
Bigrams Probabilities	Precision	0.901	0.351	0.396	0.239
	Recall	0.574	0.612	0.338	0.785
	F-measure	0.701	0.447	0.365	0.366

Tabla 2. Número de candidatos por clase.

Emoción	Votación Mayoritaria	Votación Unánime
Alegría	89	25
Tristeza	43	11
Ira	33	14
Miedo	39	13
Total	204	63

5. Conclusiones y trabajo futuro

En este trabajo se presentó la comparación del rendimiento de la selección de tweets candidatos para la retroalimentación de corpus mediante la técnica de ensamble de clasificadores Bagging utilizando tres modelos de clasificación: Ranking, Naive Bayes y Probabilidad de bigramas.

El valor de las medidas Presicion, Recall y F-measure de los modelos Ranking y Probabilidad de bigramas son cercanas entre si, mientras que las medidas

Tabla 3. Promedio de candidatos por clase.

Emoción	Promedio Votación Mayoritaria	Promedio Votación Unánime
Alegría	75	26
Tristeza	46	9
Ira	40	19
Miedo	37	13
Total	198	67

Tabla 4. Validación de corpus retroalimentado.

	Retroalomentación por votación mayoritaria			Retroalimentación por votación unánime		
Iteración	Ranking	Naïve Bayes	Probabilidad de Bigramas	Ranking	Naïve Bayes	Probabilidad de Bigramas
1	0.739	0.8	0.773	0.788	0.852	0.826
2	0.838	0.904	0.887	0.83	0.896	0.878
3	0.715	0.822	0.777	0.768	0.886	0.819
4	0.761	0.865	0.805	0.762	0.866	0.805
5	0.795	0.863	0.833	0.826	0.882	0.855
Promedio	0.769	0.851	0.815	0.795	0.876	0.836

de Naïve Bayes son mas bajas respecto a las dos anteriores. Los candidatos obtenidos por la votación mayoritaria no son confiables como se desearía puesto que sugiere una mayor cantidad de candidatos respecto a los que se tienen etiquetados en el conjunto de prueba, tal es el caso para las clases tristeza y miedo.

En la prueba de selección de candidatos donde se utiliza el corpus de entrenamiento original para los tres modelos, la cantidad de elementos seleccionados está dentro del rango del porcentaje de las pruebas que se realizaron tomando segmentos aleatorios del corpus de entrenamiento para cada modelo, por lo que se asume que los rendimientos de los clasificadores en ambos casos es similar.

Los resultados obtenidos de la validación de corpus muestran que el corpus que fue retroalimentado con los candidatos sugeridos por votación mayoritaria se vio más afectado en la integridad respecto al que fue retroalimentado con los candidatos sugeridos por votación unámine, a pesar de esto, ambos presentan una baja en cuanto a la validación original que fue del 90 %.

Como trabajo futuro, se realizarán más pruebas de retroalimentación para visualizar las modificaciones que pueda sufrir el corpus, además se implementará la técnica Bootstrapping para la complementar una retroalimentación automática del corpus integrando modelos de selección de elementos semilla para la extracción masiva de tweets que puedan ser clasificados por los modelos aquí presentados.

Referencias

1. Enríquez, F.: Técnicas de Bootstrapping en el Procesamiento del Lenguaje Natural. Memoria del Periodo de Investigación en el Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla (2007)
2. Rokach, L.: Pattern Classification Using Ensemble Methods. Series in machine perception and artificial intelligence, 75, World Scientific (2010)
3. Breiman, L.: Bagging predictors. Machine learning, 24(2), pp. 123–140 (1996)
4. Li, H.: Learning to rank for information retrieval and natural language processing. Synthesis Lectures on Human Language Technologies, 7(3), pp. 1-121 (2014)
5. Russell, S., Norving, P.: Artificial Intelligent A Modern Approach. Third Edition, Prentice Hall (2010)
6. Rodríguez, H.: Lingüística y estadística, incompatibles?. Tecnologías del texto y del habla, 72(89) (2004)
7. Basharin, G.P., Langville, A.N., Naumov, V.A.: The life and work of AA Markov. Linear algebra and its applications, 386, pp. 3–26 (2004)
8. De la Vega, M., Vea-Murguía, J.: Ensemble algorithm with syntactical tree features to improve the opinion analysis. Comité organizador, 53 (2015)
9. Carrasco, A., Portugal, R., Peralta, B.: Reconocimiento biométrico de audio y rostro: un sistema viable de identificación. Pontificia Universidad Católica de Chile Departamento de Ciencia de la Computación (2006)
10. Cadenas, J.M., Garrido, M.C., Díaz, R.A.: Soft computing en ensambles basados en boosting, bagging y random forest. Rev. Iberoam. de Sistemas, Cibern. e Informática, 7(1), pp. 25-32 (2010)
11. Hernández, J., Higuera, O., Martínez-Trinidad, J.: Detección de correo electrónico Spam usando clasificadores supervisados. Seminario Nacional de Aprendizaje e Inteligencia Computacional (2014)
12. Zhou, G., Shen, D., Zhang, J., Su, J., Tan, S.: Recognition of protein/gene names from text using an ensemble of classifiers. BMC bioinformatics, 6(1), S7 (2005)
13. Onan, A., Korukoğlu, S., Bulut, H.: Ensemble of keyword extraction methods and classifiers in text classification. Expert Systems with Applications, 5(7), pp. 232–247 (2016)