Canillita

Recuperación de Información y Recomendaciones en la Web

Informe Final

Grupo 14

Andrés Bello 4.662.672-1

Matías Sclavi 4.682.799-5

Índice

Introducción **2**

Problema **2**

Solución y arquitectura **2**

Enfoque de la solución 2

Arquitectura 3

Servidor 3

Cliente 4

Versionado y hosting del servidor 4

Scraping **4**

Funcionalidades y uso de la aplicación **5**

Experimentos y resultados **11**

Conclusiones **12**

Trabajo futuro **13**

Referencias **13**

Introducción

Como causa de la explosión de las redes sociales y los avances de las aplicaciones web y móviles las empresas encargadas de generar contenido informativo crearon plataformas propias (páginas web o aplicaciones móviles) para que los usuarios puedan consumir sus noticias. Por otro lado hacen una gran uso de las redes sociales para transmitir contenido a la enorme cantidad de usuarios que utilizan redes sociales, como por ejemplo *Twitter* o *Facebook*.

Dada el amplio abanico de posibilidades para poder ver noticias a través de la tecnología se desea poder construir una aplicación que centralice todas las noticias de varios sitios y de esta forma facilitar al usuario el consumo de las mismas.

Problema

Se pretende generar un concentrador de noticias para que no solo sea más cómodo a la hora de informarse (no deba visitar varias webs), sino que también para que amplíe el universo de búsqueda a aquellas personas que no conozcan ciertos sitios o no esté al tanto de las últimas noticias, pueda acceder a esta información. Para ello se obtienen las noticias de los principales diarios de Uruguay y se procesa la información para luego poder presentarla en una aplicación móvil junto con la posibilidad de realizar filtros de búsqueda. Los diarios y portales seleccionados para realizar una primera versión de la aplicación fueron: "El País", "El Observador", "180", "Espectador", "La Red 21", "La Diaria", "Montevideo Portal", "El Diario y La República".

La solución también intenta resolver la oportunidad de poder filtrar noticias por categorías (probablemente el filtro más utilizado por los usuarios). Para poder resolver esto se debe encontrar un camino para poder determinar la categoría de las noticias de aquellos portales que no las especifican, y aunque estén especificadas se deben poder catalogar dentro de una cantidad de clases conocidas y no crear categorías de forma dinámica y redundante. La solución para esto es un clasificador de noticias implementado con técnicas de aprendizaje automático.

Solución y arquitectura

Enfoque de la solución

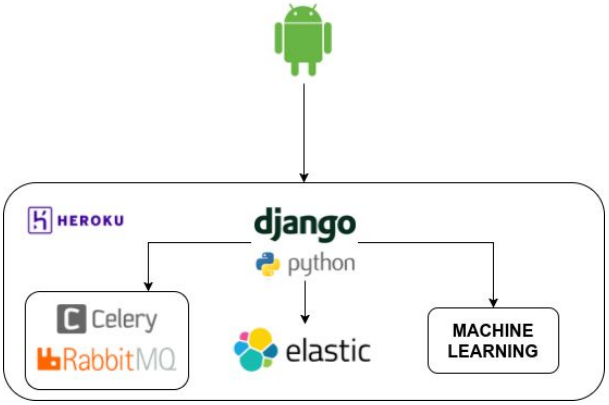
La solución que se propone es utilizando la técnica de *scraping* para poder obtener las noticias de los diferentes portales, luego una vez que se obtuvieron las noticias se almacenan en *Elasticsearch*[1] para poder utilizar su implementación de índices invertidos para poder implementar filtros de búsqueda sobre diferentes atributos, como por ejemplo: títulos, cuerpo de las noticias, fecha, categoría, etc. Por otro lado se construirá un clasificador que etiquete las noticias con una categoría conocida y luego poder aplicar filtros sobre las mismas. Por último se implementará una aplicación móvil para que los usuarios puedan utilizar las funcionalidades. Resumiendo la solución consiste en los siguientes puntos:

- Servidor para implementar el *scraping* y contenga una capa de servicios para que utilice la aplicación móvil.
- *ElasticSearch*

- Clasificador de categorías
- Aplicación Móvil

Arquitectura

A continuación vamos a presentar una arquitectura a alto nivel del sistema a construir:



Se plantea una arquitectura cliente-servidor, en donde el cliente es una aplicación *Android* y el servidor está construido con *framework* de *Python* llamado *Django*[2] y otra serie de plataformas que se describirán a continuación.

Servidor

PYTHON + DJANGO

Este componente tiene la responsabilidad de realizar *scraping* sobre los portales de los diarios mencionados en la sección anterior para luego procesarlos y almacenarlos en *Elasticsearch*. Por otro lado brinda las *APIs REST* para que el cliente pueda obtener las noticias que debe mostrar al usuario y a la vez pueda realizar búsquedas en las mismas. También ofrece *APIs* para poder obtener las noticias recomendadas para el usuario y las noticias que son tendencias en ese momento. Se decidió utilizar esta tecnología dado que el *framework* es dedicado a trabajar con *REST*.

ELASTICSEARCH

Se utiliza esta plataforma para almacenar las noticias y poder realizar búsquedas eficientes sobre ellas, *Elasticsearch* implementa la técnica de índice invertido sobre documentos, lo cual es de suma utilidad para poder implementar filtros sobre los cuerpos de las noticias, títulos y fechas.

MACHINE LEARNING

El servidor va a tener un motor de inteligencia artificial que sea capaz de catalogar las noticias. La idea es poder darle la posibilidad al usuario de poder filtrar por la categoría de la noticia. El problema con ello es que no todos los diarios manejan el mismo conjunto de categorías lo que hace muy difícil unificarlas dentro de un solo criterio. Por lo tanto para resolver este problema debemos generar un corpus de noticias etiquetadas según su categoría para así entrenar un clasificador tal que, dada una noticia, se obtenga la categoría asociada. De esta forma logramos brindar al usuario la posibilidad de filtrar noticias por un conjunto de categorías universal independiente del diario dueño de la noticia. Para poder lograr esto utilizamos la librería scikit-learn de Python que provee diferentes soluciones sobre este tema.

RABBIT MQ + CELERY

El scraping de las noticias es una tarea que se debe hacer de forma asincrónica todos los días a una determinada hora para así no disminuir la performance y la usabilidad de la aplicación cuando el usuario la esté utilizando. Para ello vamos a utilizar las plataformas *Rabbit*[3] y *Celery*[4]. Estas dos en conjunto permiten especificar tareas y programar a qué hora/día se deben ejecutar.

Cliente

ANDROID

El cliente es el responsable de presentar las noticias y demás funcionalidades brindadas por el servidor. La aplicación se desarrollará en *Android* nativo para que una mejor performance de la que se podría lograr a través de distintos *frameworks* que utilizan otros lenguajes y luego son compilados para distintas plataformas.

Versionado y *hosting* del servidor

Para facilitar el desarrollo del servidor, se utilizan las herramientas *BitBucket*[5] y *Heroku*[6], siendo la primera un repositorio *Git* al cual subimos nuestro código fuente y la segunda una plataforma en la nube que permite tener un servidor funcionando, así no hay que ejecutarlo de manera local. Estas dos últimas plataformas se integran entre sí para que cuando se sube una nueva versión del código al repositorio en Bitbucket este actualice el código almacenado en el servidor de Heroku.

Scraping

El *scraping* se realiza con la librería *BeautifulSoup 4*[6] (*BS4*) perteneciente a *Python*, la cual fue seleccionada luego de buscar entre varias herramientas para esta tarea. Fue seleccionada debido a las siguientes características:

- ***Open Source*:** se cree relevante utilizar herramientas *Open Source*, ya que en la mayoría de estas podemos encontrar una mejor documentación de la herramienta y una mayor gama de experimentos realizado por otros usuarios en web, entre otros.
- **Tipos de consulta:** nos permite realizar búsquedas y navegaciones idiomáticas sobre el *HTML* de las páginas.
- **Performance:** si bien no tiene el mismo rendimiento que el *parser* sobre el que está implementado *BS4*, tiene mejor *performance* que otros *parsers*. Permite también mejor rendimiento al *parsear* una parte del documento en vez de este entero.

El *scraping* se realiza en dos tareas: una para las portadas y otra para los diarios en general. Esto se hace de esta manera para poder obtener las portadas cada menor cantidad de tiempo, ya que es muy común que los usuarios entren a los portales de noticias a ver las noticias más importantes. Dado esto, se intenta brindar la información lo más actualizada posible al usuario.

PORTADAS

El *scraping* de las portadas es el más sencillo: para cada diario obtenemos el *HTML* de su portal y obtenemos la noticia principal. Para saber que objetos obtener dentro del *HTML* para leer la portada se realiza una inspección a través de las *Herramientas para desarrolladores* que brinda *Google Chrome*, o cualquier otra herramienta de un navegador que permita inspeccionar la web que estamos visualizando. De estos objetos obtenemos el link a la noticia, la imagen y el título de la misma. El segundo paso es obtener el *HTML* interior de la portada a partir del link obtenido previamente. Luego hay que realizar un procedimiento muy parecido al descrito anteriormente para obtener el cuerpo de la noticia.

NOTICIAS POR DIARIO

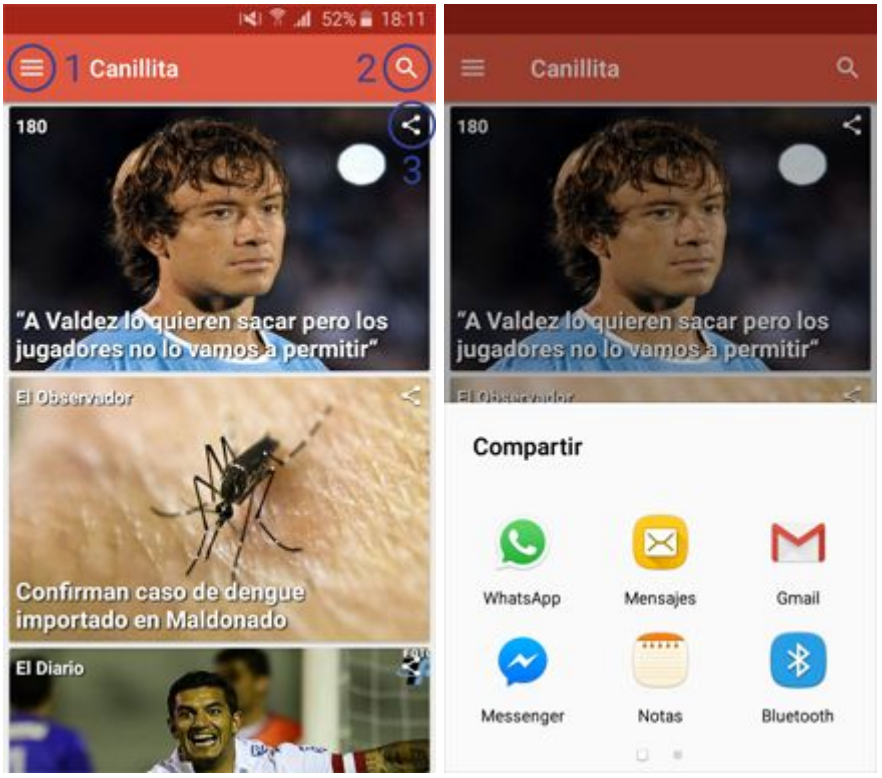
Para cada diario se realizó un procedimiento muy parecido al de las portadas, solo que se obtienen como máximo diez noticias de cada portal por ejecución de la tarea programada en vez de una sola. Luego de obtenerlas, hay que iterar sobre estas para poder obtener la información necesaria para nuestro sistema.

Funcionalidades y uso de la aplicación

Como se explicó en la sección anterior la aplicación mostrará las noticias (portadas de la misma) más importantes de los diarios que se seleccionen dentro de los nombrados en la primer sección. Por otro lado, podrá ingresar a cualquiera de los diarios de la aplicación y ver las noticias de ese diario. El usuario también podrá optar por realizar una búsqueda de noticias, utilizando una combinación entre: palabras claves que le interesen buscar tanto en el título como en el cuerpo de la noticia, a partir de una fecha, hasta una fecha y ciertas categorías.

Profundizando un poco más en el tema de las categorías, tuvimos que elegir entre las distintas categorías que tienen los diarios. Estas fueron elegidas a partir de: “El Observador”, “La Red 21” y “Espectador”. ¿Por qué se tomaron sólo las categorías de tres diarios y no de todos? Se realizó de esta manera porque se intentará predecir cuales son las categorías de las noticias de los otros diarios a partir de su contenido. La razón detrás de esto es que no todos los diarios tienen secciones y sería bueno poder categorizar estas noticias para una búsqueda por temas. Otra razón es que no todos los diarios tienen el mismo criterio a la hora de categorizar sus noticias, por lo que puede parecerle raro al usuario de que una noticia pertenezca a una categoría en un diario y a otra en otro diario. La predicción será hecha con un algoritmo de aprendizaje automático, por lo que se debe crear un corpus anotado de noticias a partir de los diarios seleccionados para obtener las categorías. Esto nos servirá de investigación, particularmente que tan difícil es crear un buen corpus anotado y ver con que cantidad de ejemplos se puede tener una buena performance del clasificador.

INICIO



En la pantalla de Inicio podemos encontrar las portadas de los diarios que fueron utilizados en la recopilación de noticias. Podemos también compartir estas noticias vía distintas vías, como también podemos realizar una búsqueda sobre las noticias que hemos recopilado. Otra cosa que podemos ver es el menú principal, el cual está en todas las vistas de la aplicación.

1. **Menú:** aquí podemos encontrar las distintas secciones de la aplicación.
2. **Búsqueda:** al tocar en la lupa se nos despliega el menú para poder realizar una búsqueda entre las noticias. Esta lupa la podemos encontrar en todas las vistas de la aplicación.
3. **Compartir:** podemos compartir cualquier noticia que queramos a partir de este botón. Vemos el menú que se nos despliega en la segunda imagen.

MENÚ

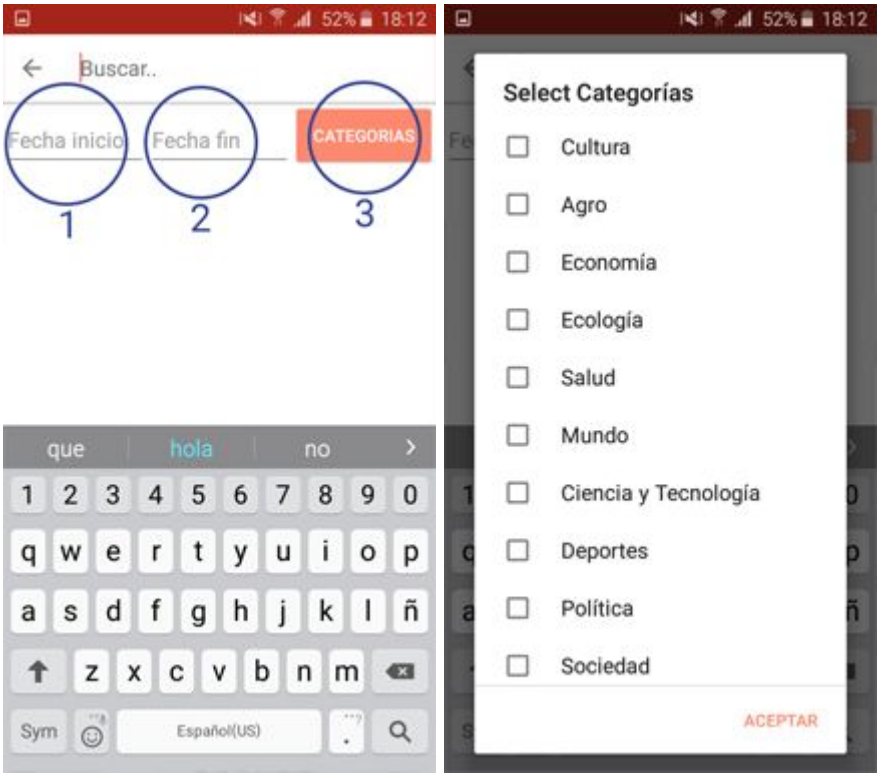
En el menú podemos encontrar las distintas secciones de la aplicación. Podemos cambiar entre la vista de las portadas de los diarios, como ver las noticias de cada usuario.

1. **Inicio:** podemos ir a la vista de las portadas de los diarios.
2. **Diarios disponibles:** al tocar este *item*, podemos encontrar todos los diarios para los cuales se obtuvieron noticias.
3. **Diario seleccionado:** desde este menú desplegable podemos acceder a cada uno de los diarios para poder ver noticias exclusivas de este.
4. **Habilitar/deshabilitar portada:** este slide le permite al usuario habilitar/deshabilitar la portada de un determinado diario en la vista inicial.

DIARIO INDIVIDUAL

En esta vista podemos encontrar todas las noticias para el diario seleccionado a partir del menú. Como podemos ver, se puede identificar el diario para el cual estamos viendo las noticias en la parte superior de la vista. Además, podemos realizar las acciones de compartir la noticia y realizar una búsqueda, al igual que en la vista de las portadas. Esta búsqueda no es restringida a las noticias del diario actual.

BÚSQUEDA



Al tocar sobre la lupa, podemos realizar distintos tipos búsquedas sobre las noticas de nuestro sistema. La forma principal de realizar la búsqueda es a través de ingresar una *query* en el campo principal de búsqueda. Esta *query* se busca sobre el cuerpo y el título de la noticia. Luego podemos realizar distintos tipos de combinación de filtrados sobre las noticias, donde podemos optar entre los siguiente filtros:

- A partir de cierta fecha ingresada.
- Previo a cierta fecha ingresada.
- Seleccionar las categorías sobre las cual buscar.

Una vez que el usuario haya ingresado los parámetros deseados para la búsqueda, se realiza la misma al presionar el botón con la lupa en nuestro teclado.

1. **Fecha de inicio:** campo para ingresar la fecha a partir de la cual se quieren buscar las noticias.
2. **Fecha de fin:** campo para ingresar la fecha hasta la cual se quieren buscar las noticias.
3. **Categorías:** al tocar sobre este botón de le despliega un *pop-up* al usuario en el cual puede seleccionar una o más categorías en las cuales buscar las noticias.

Una vez que se realiza la búsqueda, vamos a poder ver las noticias que cumplen con los parámetros ingresados por el usuario. Al tocar una de las noticias de esta lista nos enviará a la vista de la noticia. No es necesario esto último, pudiendo cambiar los parámetros de la búsqueda y comenzar de nuevo.

NOTICIA

Al tocar sobre una noticia (tanto en la vista de portadas, como en la de cada diario o en la búsqueda) el usuario podrá ingresar a la información de la noticia. En esta se puede ver el nombre del diario a la cual pertenece, la imagen principal de la noticia, el título y el cuerpo de la misma. Como en las otras vistas, podemos ingresar tanto a el menú principal como el de la búsqueda.

Experimentos y resultados

Ocurrió un problema a la hora de la recolección de noticias: se alcanzó la cuota de almacenamiento de la herramienta *Elasticsearch* previo a lo esperado. Esto nos dificulto a la hora de realizar el entrenamiento de los métodos de aprendizaje automático, ya que en cada entrenamiento en vez de entrenar con la cantidad total de noticias procesadas por las tareas de *scraping*, se entrena con las que estén almacenadas en *Elasticsearch* en ese momento.

La primera solución que se ideó para solucionar este problema fue utilizar un método de aprendizaje *online*, los cuales son entrenados en forma secuencial con distintos conjuntos de datos para ir actualizando el clasificador. El problema con esto es que las *features* del clasificador va variando con el *set* de entrenamiento, por lo que al realizar una iteración del entrenamiento, el clasificador no tendrá las mismas características con las que entrenar. La segunda manera que se encontró para sortear este problema fue una guardar en un archivo los campos más importantes de las noticias, es decir, su título, su cuerpo y su categoría. Cada vez que se ejecutaba una tarea de *scraping* se añaden los campos de las noticias por las tareas. Con este archivo podemos entrenar un clasificador *batch*, como se deseaba en un principio.

PRE PROCESAMIENTO

Para entrenar el clasificador con las noticias recabadas, previamente se debe hacer un pre procesamiento de las mismas para no generar ruido a la hora de entrenar, por ejemplo, para entrenar el clasificador no es necesario que las noticias contengan los *links* hacia portales externos (*Twitter, Facebook, etc.)*. También es importante realizar un procesamiento del lenguaje natural de las noticias, en este caso se decidió remover las *stop words* de las noticias, los *links*, y el código html/Javascript que se puede encontrar dentro del cuerpo de las noticias.

CARACTERÍSTICAS

El primer método utilizado para la obtención de características fue *CountVectorizer* provisto por la librería, el cual implementa la técnica "Bolsa de palabras". Sabiendo que con este método no íbamos a extraer la información suficiente se decidió utilizar otras técnicas:

- n-gramas de palabras: una y dos palabras.
- n-gramas de caracteres: entre uno y cinco caracteres.
- stems.

CLASIFICADOR Y PRIMEROS RESULTADOS

Para realizar los entrenamientos se utilizó el clasificador *Multinomial NB* que implementa esencialmente un clasificador Bayesiano, basándose en las frecuencia de ocurrencias. Si bien este clasificador no es el más sofisticado, es muy utilizado para dar una primera idea de los resultados que se pueden llegar a obtener. Para esta tarea vamos a seguir la siguiente metodología: de las noticias etiquetadas tomaremos el 80% de ellas para el entrenamiento y el 20% restante para la evaluación del clasificador. Sobre el conjunto de entrenamiento, para obtener resultados consistentes, se utilizara validación cruzada.

Los primeros resultados obtenidos fueron más o menos los que se esperaban de acuerdo a la cantidad de noticias obtenidas en ese momento (primer y segundo día de recopilación de noticias), a continuación se pueden observar estos resultados:

#noticias de entrenamiento	Precisión
30	0,32
85	0,46

ENTRENAMIENTO FINAL

Luego de realizar los primeros experimentos para entrenar el clasificador se utilizó el conjunto de noticias que se fueron almacenando en el archivo descrito previamente el cual contiene aproximadamente 1800 noticias etiquetadas según su categoría, por otro lado se recopilaron aproximadamente 5500 noticias sin etiquetar. El *corpus* de noticias etiquetadas quedó dividido de la siguiente forma:

Entrenamiento	Evaluación
1440	400

Los resultados obtenidos en el último entrenamiento registrado se muestran a continuación:

#noticias de entrenamiento	Precisión
1440	0,73

Si bien estos no son los resultados que uno quisiera tener, son esperables dada la cantidad de noticias recopiladas. Para las noticias que no fueron etiquetadas, su categoría será generada a partir de este clasificador.

Conclusiones

Una dificultad a la hora de hacer *scraping* sobre los diarios fue la diversidad con la que los portales publican las noticias. No solo eso, sino que un cambio en una web repercute en el código utilizado para obtener la información. Una posible solución sería tener una forma estándar de maquetado para facilitar la tarea que realizamos. Otra solución posible sería que tengan *APIs* públicas para que no se tuviera que realizar el *scraping* y se pudiera obtener la información solo con una consulta. Sabemos que esto es casi improbable, ya que cada sitio compite con los demás, pero si se lograra, podría haber una mayor accesibilidad a la información para la población.

Siguiendo con la línea anterior, el hecho de implementar una aplicación móvil permite no solo un centralizador de noticias, sino también contar con la posibilidad de poder compartir a distintas aplicaciones de celulares, con los cuales estamos en constante contacto a lo largo del día. Dado esto, si las personas compartieran noticias relevantes o destacadas del día, tendríamos un flujo de información constante.

Con respecto al clasificador, dado el problema que ocurrió con la herramienta de almacenamiento, no se pudo tener los resultados esperados. Lo importante que sacamos de esta experiencia fue la formulación de un *corpus* propio y la soluciones planteadas para sortear este problema.

Trabajo futuro

Debido a los resultados obtenidos y las posibilidades tecnológicas nos proponemos mejorar el clasificador de categorías mejorando el corpus de entrenamiento y la técnica de aprendizaje automático. Para mejorar el corpus podría construirse uno con mayor cantidad de noticias etiquetadas, para ello podemos obtener noticias de diferentes fuentes y no solamente de las plataformas de cada diario, sino también utilizando el contenido que se publica en redes sociales, como por ejemplo *Twitter*. Una forma de mejorar el clasificador es utilizando técnicas más avanzadas, como por ejemplo, redes neuronales o *deep learning* utilizando librerías como *TensorFlow*[7] y *Keras*[8], o investigando la existencia de una red ya entrenada y adaptándola a nuestra solución. Por último se propone mejorar el pre procesamiento del corpus de noticias para poder quitar posible ruido que genere un mal comportamiento de los clasificadores.

Por otro lado se propone mejorar la *performance* y exactitud las de las búsquedas realizadas mediante *Elasticsearch* mediante un mayor entendimiento de la solución y las diferencias entre todos los posibles caminos de búsqueda que brinda.

Por último se quiere ampliar el abanico de noticieros de los cuales se extrae noticias para que no sea solamente una solución local y también construir la misma aplicación para las diferentes plataformas, por ejemplo, *iOS* y una aplicación web.

Referencias

1. Elasticseach: https://www.elastic.co/products/elasticsearch
2. Django: https://www.djangoproject.com/
3. RabbitMQ: https://www.rabbitmq.com/
4. Celery: http://www.celeryproject.org/
5. BitBucket: https://bitbucket.org/
6. Heroku: https://www.heroku.com/
7. TensorFlow: https://www.tensorflow.org/
8. Keras: https://keras.io/