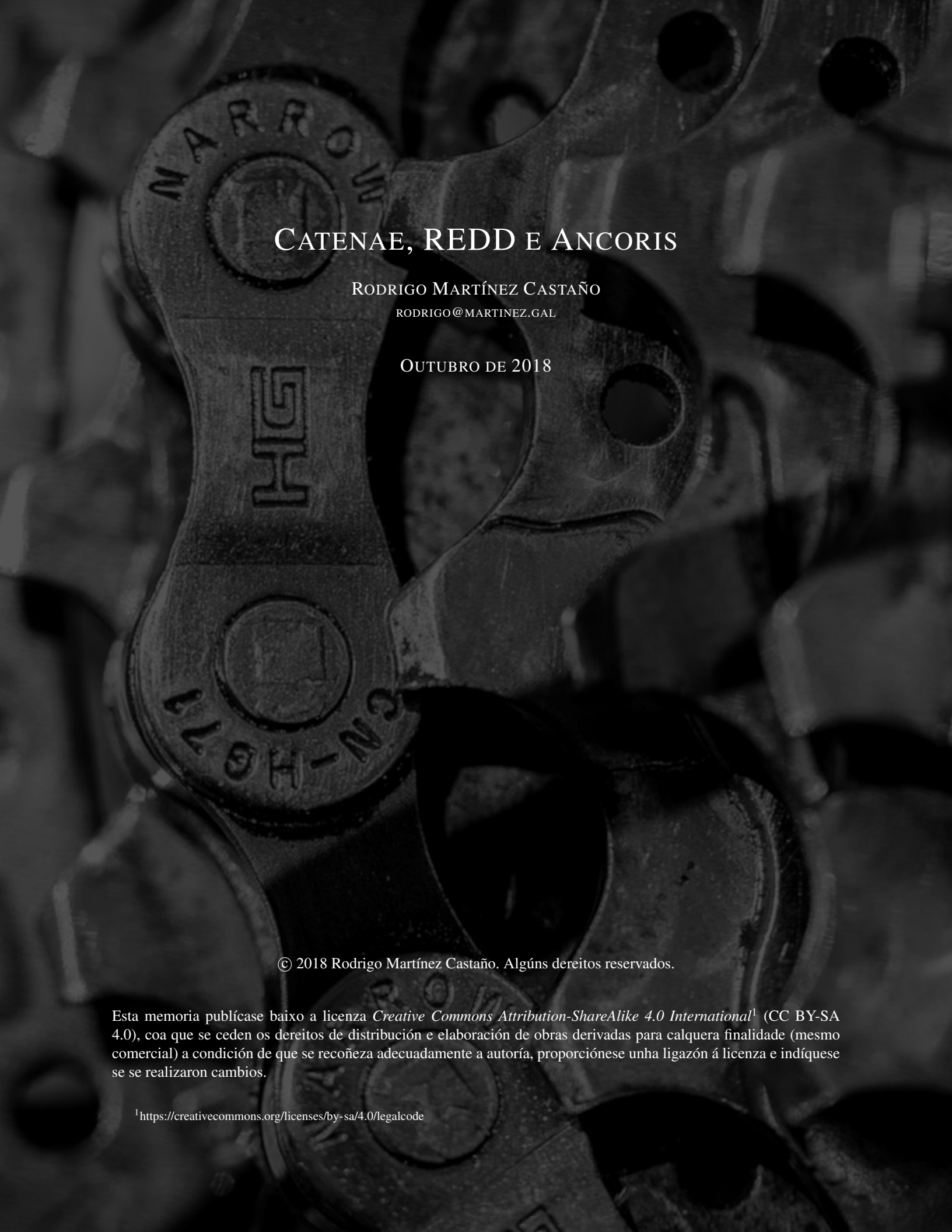

CATENAE, REDD E ANCORIS

RODRIGO MARTÍNEZ CASTAÑO
RODRIGO@MARTINEZ.GAL

OUTUBRO DE 2018

Índice

Resumo

Nesta memoria preséntanse tres achegas. En primeiro lugar (1) Catenae, unha libraría Python que permite construír de forma sinxela sistemas modulares e escalables de procesamento en tempo real. A libraría foi deseñada para servir de base a (2) REDD (*REddit Depression Detection*) pero cun enfoque amplo, buscando poder converter rapidamente un prototipo escrito en Python (de calquera índole) nun sistema escalable de procesamento en tempo real. REDD é unha plataforma integral para a detección temperá de risco de depresión en tempo real que opera sobre Reddit e na que os expertos poden revisar as alertas xeradas. Por último, (3) Ancoris é un xestor de recursos para clústers que, de novo, está ideado desde unha perspectiva xenérica (xestión sinxela de recursos dun clúster a través de contedores Docker) pero cunha aplicación concreta: prover os recursos necesarios para a execución de sistemas creados con Catenae.

1. Introdución

Nos últimos anos desenvolvéronse numerosas tecnoloxías destinadas ó procesamento de datos masivos, moitas delas de código aberto e de uso libre. Estas plataformas céntranse na escalabilidade horizontal, o que implica que para o procesamento dunha maior cantidade de datos sen grandes distorsións no ritmo, non é necesario aumentar ou actualizar os recursos dunha máquina (escalabilidade vertical), senón que é suficiente con engadir máis nodos con similares características a un clúster. A proliferación deste tipo de tecnoloxías de código aberto democratizaron e condicionaron o gran número de aplicacións que fan uso destas plataformas en multitude de ámbitos, tanto profesionais como académicos.

Centrándonos nos *frameworks* de procesamento, atopámonos cunha importante limitación: os datos teñen que poder dividirse en grupos independentes, de tal xeito que sexa posible paralelizar o traballo en diferentes máquinas aínda que existan puntos de procesamento secuencial. Existen dous grandes tipos de tecnoloxías de procesamento deste tipo: procesamento de lotes (*batch processing*) e procesamento de fluxos (*stream processing*).

No primeiro caso, os resultados finais obtéñense xuntos ó finalizar o procesamento do lote de datos composto por unha ou máis etapas. Para definir o traballo a realizar, defínese unha topoloxía de procesamento que indica o fluxo dos datos a través das distintas etapas. Cada nodo (físico ou virtual) pode executar unha instancia da topoloxía (illada do resto de instancias), repartíndose os datos de forma equitativa entre as instancias existentes. Nas tecnoloxías de procesamento de fluxos, as distintas etapas dunha topoloxía son independentes e non pertencen a unha instancia concreta. Por tanto, as distintas etapas poden ser paralelizadas de forma individual sen aumentar o grao de paralelismo de toda a topoloxía. Estas tecnoloxías son adecuadas para aplicacións que obteñen información en tempo real e deben dar unha resposta inmediata, xa que cando un dato completa o seu camiño a través das distintas etapas, o resultado pode obterse de forma instantánea. Por outra banda, co procesamento de lotes os resultados obtéñense cando un lote de datos é procesado por completo. Un caso de aplicación de procesamento en tempo real é a análise de contidos en redes sociais para a detección temperá de riscos. Este será o obxectivo principal deste proxecto.

A sociedade está exposta a un gran abanico de riscos e ameazas, moitos dos cales se exteriorizan en Internet a través das redes sociais. Algúns dos riscos son externos, xerados por criminais, acosadores, etc. Outros, por outra banda, orixínanse polos propios usuarios. Por exemplo, a depresión pode levar a desordes alimenticias como a bulimia ou a anorexia ou mesmo ó suicidio [1]. A detección temperá de riscos consiste en descubrir situacións con probabilidade de evolucionar negativamente cara a escenarios de maior gravidade. Unha detección temperá apropiada pode reducir ou minimizar problemas. Algúns tipos de riscos poden ser detectados analizando a actividade de usuarios en Internet. Actualmente, as tecnoloxías existentes son maiormente reactivas: as alertas adoitan dispararse cando os problemas xa apareceron. A detección temperá de riscos é unha área de estudo cuxa importancia está en auxe.

Neste traballo preséntanse tres achegas. En primeiro lugar (1) CATENAE[2], unha libraría Python que permite construír de forma sinxela sistemas modulares e escalables de procesamento en tempo real. A libraría foi deseñada para servir de base a (2) REDD[3] (*RE*ddit *D*epression *D*etection) pero cun enfoque amplo, buscando poder converter rapidamente un prototipo escrito en Python (de calquera índole) nun sistema escalable de procesamento en tempo real. REDD é unha plataforma integral para a detección temperá de risco de depresión en tempo real que opera sobre Reddit e na que os expertos poden revisar as alertas xeradas. Por último, (3) ANCORIS[4] é un xestor de recursos para clústers que,

[2]Dispoñible en https://github.com/catenae

[3]Dispoñible en https://github.com/redd-system

[4]Dispoñible en: https://github.com/ancorisrm

de novo, está ideado desde unha perspectiva xenérica (xestión sinxela de recursos dun clúster a través de contedores Docker [2]) pero cunha aplicación concreta: prover os recursos necesarios para a execución de sistemas creados con CATENAE.

Reddit é un sitio web onde os usuarios publican contidos como textos, imaxes ou ligazóns, e onde outros usuarios poden comentar e votar a favor ou en contra. A plataforma subdivídese en distintas comunidades centradas en temas concretos. Actualmente, Reddit está situado como o sexto sitio web con maior tráfico do mundo segundo a clasificación de Alexa[5]. A media de usuarios mensuais é superior ós 330 millóns e existen máis de 138.000 comunidades activas[6].

Tanto CATENAE como REDD foron presentados en dous congresos de investigación: *European Conference on Information Retrieval* [3] e *Spanish Conference in Information Retrieval* [4].

2. Aspectos técnicos

2.1. Catenae

A libraría CATENAE desenvólvese ante a necesidade de proporcionar unha base para a construción de topoloxías para a plataforma REDD en Python. Escóllese esta linguaxe por dúas razóns principais: a súa popularidade crecente en Ciencia de Datos e a existencia de librarías importantes como scikit-learn [5], empregada ademais para a construción do clasificador orixinal de usuarios en redes sociais.

Actualmente, o *framework* máis popular para procesamento de fluxos de datos, Apache Storm [6], conta cun soporte para Python pouco eficiente. Ó non ser unha linguaxe JVM, Storm comunícase cos procesos Python indirectamente. Ademais, a xestión de dependencias é un proceso tedioso á hora de despregar topoloxías.

CATENAE proporciona unha contorna para desenvolver sistemas modulares de procesamento en tempo real, delegando o paso de mensaxes entre módulos a Apache Kafka [7]. Ademais, isto permite conectar as topoloxías creadas con CATENAE con outros sistemas de forma sinxela. O obxectivo desta libraría é facilitar a escalabilidade horizontal de aplicacións escritas en Python a través da división do procesamento en micromódulos (etapas de transformación ou filtrado dunha topoloxía) que se executarán dentro de contedores Docker, proporcionando illamento entre módulos. Debido a esta característica, é posible utilizar distintas versións dunha libraría en distintos módulos dunha mesma topoloxía e mesmo distintas versións do intérprete de Python. Algunhas características destacables son a serialización transparente e o sistema de priorización de fluxos de entrada para os módulos. As topoloxías pódense escalar automaticamente executando máis instancias dun determinado módulo ou destruíndoas.

Existen tres tipos de micromódulos en CATENAE (ver Figura 1):

- Micromódulos fonte. Estes módulos non reciben datos de Kafka, senón que recupéranos de APIs externas, bases de datos ou mediante *crawling* web, por exemplo.
- Micromódulos intermedios. Todos os módulos que transforman os datos de entrada e/ou os filtran. Consumen e emiten datos dentro da topoloxía.
- Micromódulos finais. Estes módulos consumen datos da topoloxía pero non os emiten de volta. Almacenan os datos de forma personalizada en bases de datos ou ficheiros.

CATENAE soporta múltiples conexións entrantes para un determinado nodo da topoloxía. Existen dous modos implementados para xestionar estas situacións. O modo por defecto permite procesar os datos recibidos asignando a mesma prioridade a todas as fontes de entrada. No segundo modo, asígnase a cada fonte un tempo que vai crecendo de forma exponencial coa prioridade dentro dunha xanela temporal.

A libraría tamén permite crear e borrar conexións entre os módulos en tempo de execución. Ademais é posible executar certas partes dunha topoloxía de forma síncrona e executar funcións *callback* cando un dato saínte é correctamente rexistrado.

Os módulos dunha topoloxía poden emitir calquera obxecto Python, que será serializado e comprimido de forma automática. O módulo que o reciba obterá o obxecto deserializado automáticamente.

[5] https://www.alexa.com/siteinfo/reddit.com/
[6] https://www.redditinc.com/

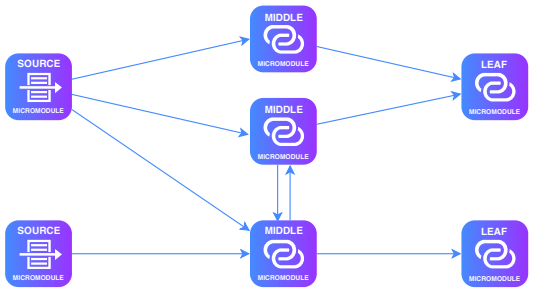

Figura 1: Diagrama de exemplo dunha topoloxía Catenae.

Os módulos despléganse no interior de contedores e, por tanto, recursos como CPU ou memoria poden restrinxirxe para cada instancia dun módulo.

Por último, CATENAE permite ó usuario cargar recursos externos durante a inicialización dos módulos. Aerospike [8] é un sistema de almacenamento distribuido clave-valor que usa a memoria RAM e discos SSD como dispositivos de almacenamento. A libraría incorpora un conector de forma nativa para poder obter os recursos necesarios de forma veloz nas inicializacións dos módulos.

2.2. REDD

A plataforma REDD permite a detección temperá de usuarios en risco de depresión en tempo real. O sistema traballa sobre a rede social Reddit cun *crawler* propio e toma como base un clasificador e un conxunto de datos de adestramento detallados en [9] para a fase de predición. A plataforma conta cos seguintes módulos: (1) unha topoloxía CATENAE para a extracción de usuarios e contidos de Reddit, (2) unha segunda topoloxía CATENAE para o clasificador de textos dos usuarios, (3) unha API HTTP para a xestión de alertas e o acceso ós textos dos usuarios, e (4) unha interfaz web para a xestión de alertas por parte dos expertos.

2.3. Topoloxía de extracción de datos de Reddit (*crawler*)

O primeiro obxectivo da plataforma é maximizar o número de usuarios rastrexados para recuperar novas publicacións de forma periódica. Para logralo, construiuse un *crawler* web para Reddit seguindo as regras expresadas no arquivo *robots.txt*. Empréganse a libraría CATENAE xa que está composto por múltiples micromódulos interconectados (ver Figura 2):

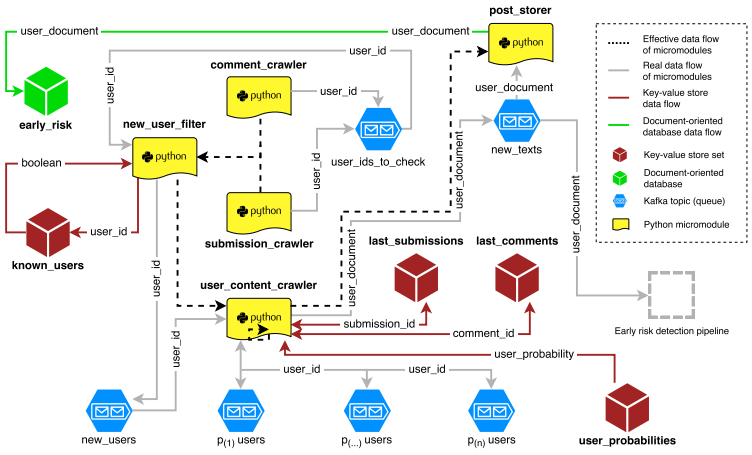

Figura 2: Diagrama de arquitectura do *crawler* de Reddit.

– **Crawlers de publicacións e comentarios**. Recuperan as novas publicacións e comentarios e extraen os nomes de usuario.

– **Filtro de novos usuarios.** Os nomes de usuarios fíltranse para evitar duplicados. Aerospike emprégase para realizar esta tarefa de forma eficiente xa que proporciona un almacenamento baseado en memoria. Os usuarios que pasen o filtro enviaranse á cola de novos usuarios.

– **Crawlers de contido.** Nesta etapa, todos os textos escritos polos usuarios que pasaron o filtro son extraídos. Os usuarios selecciónanse de distintos fluxos de entrada, ordeados por prioridade. Esta prioridade establécese en función da puntuación obtida polo clasificador en iteracións anteriores e a actividade realizada en Reddit polos usuarios dende a última vez que dito usuario foi procesado. Os textos envíanse a través dun único fluxo de saída.

– **Almacenamento dos textos.** Os textos almacénanse nunha base de datos orientada a documentos. Ademais, os textos continúan cara a topoloxía de predición.

2.4. Topoloxía de detección temperá de depresión

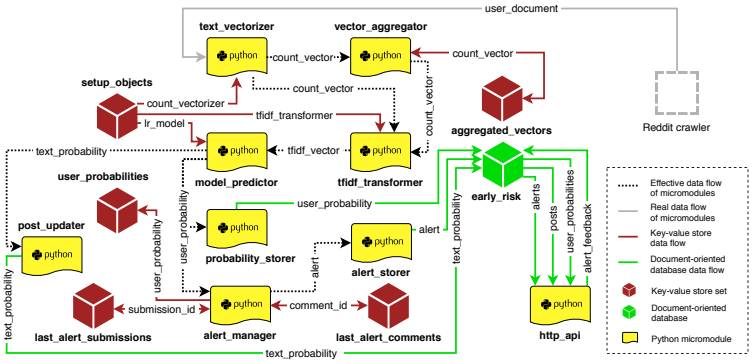

Figura 3: Diagrama de arquitectura simplificado da detección temperá de risco de depresión.

A topoloxía de detección está composta por un clasificador de regresión loxística con regularización L1 implementado con *scikit-learn*. Construiuse cun conxunto de 486 usuarios (83 positivos, 403 negativos) [9]. Os usuarios represéntanse cun único documento formado pola concatenación de todos os seus textos. O proceso de predición está composto por tres micromódulos (ver Figura 3):

– **Vectorizador de textos**. Transforma o texto de entrada nun vector de ocorrencias de *tokens*.

– **Agregador**. Agrega un vector de ocorrencias co vector agregado de ocorrencias dispoñible do usuario.

– **Transformador tf–idf**. Transforma o vector agregado nun vector normalizado cunha representacion tf-idf.

– **Predictor**. Proporciona a probabilidade de risco de depresión dun usuario dada a representación dos seus textos a través dun vector tf-idf. Tamén proporciona a probabilidade individual de cada texto que será almacenada polo módulo de almacenamento.

O módulo de almacenamento aliméntase das probabilidades de risco de depresión dos usuarios e as almacena na base de datos documental. O xestor de alertas recibe a mesma entrada de datos e almacena as alertas se a probabilidade está por riba dun determinado umbral.

A interfaz web está composta por tres vistas principais, conectadas á API HTTP. En “Alerts” (Figura 4), as alertas xeradas polo sistema poden ser inspeccionadas en tempo real, ordenándoas por tempo ou prioridade. As alertas correspóndense cun usuario determinado e todos os textos relacionados con dita alerta poden consultarse por orden descendente de probabilidade. Deste xeito, o revisor pode etiquetar as alertas como correctas ou como falsos positivos. Para cada alerta, tamén se representa a evolución do risco de depresión do usuario afectado. Nesta vista tamén existen tres pestanas. A pestana por defecto permite visualizar as alertas non etiquetadas. As outras dúas conteñen as alertas

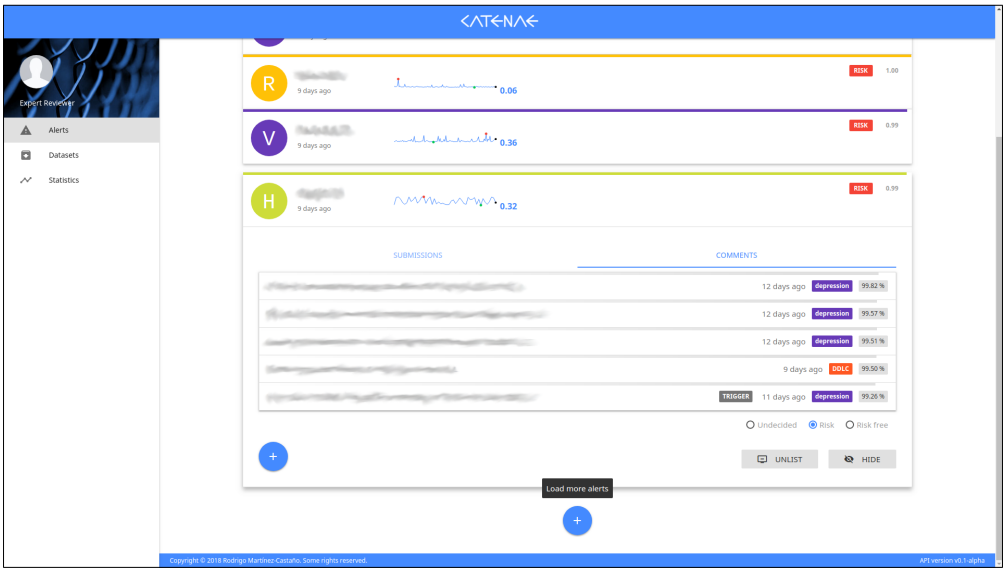

Figura 4: Vista de alertas de risco de depresión xeradas en tempo real da plataforma REDD.

etiquetadas (correctas e falsos positivos). A vista "Datasets" contén tres opcións de descarga dos distintos conxuntos de datos: alertas correctas, alertas incorrectas e todas as alertas. Deste xeito facilítase a xeración de colleccións etiquetadas. Por último, a vista "Statistics" amosa información en tempo real sobre o rendemento do sistema: número total de textos e usuarios recuperados, textos e usuarios analizados por unidade de tempo, etc.

2.5. Ancoris

Para a execución da plataforma REDD, outras topoloxías creadas con CATENAE e, en xeral, de sistemas formados por módulos empaquetados en contedores, desenvólvese ANCORIS, un xestor de recursos baseado en contedores Docker enfocado na sinxeleza de uso e nunha xestión de recursos de alta granularidad.

Ancoris permite xestionar os recursos dispoñibles nun clúster e asignalos a contedores Docker. O propio xestor execútase dentro de contedores Docker e está composto por dous tipos de procesos que poden coexistir na mesma máquina: *masters* e *workers*. Os *masters* xestionan os recursos dispoñibles no clúster e son os responsables de lanzar contedores con recursos asignados a través da API dos *workers*. Os *workers* expoñen os recursos dispoñibles do seu *host* cando son iniciados por primeira vez. Tanto as peticións externas dun cliente como as internas entre *masters* e *workers* realízanse a través das súas APIs HTTP.

O cliente interacciona co xestor de recursos a través dun nodo *master*. O proceso *master* é responsable de comprobar as solicitudes e atopar un nodo cos recursos necesarios. Se a operación é exitosa, o proceso *master* interactuará co proceso *worker* do nodo escollido para lanzar a tarefa cos recursos solicitados.

Consul [10] é un sistema distribuído de alta dispoñibilidade que proporciona unha contorna para o descubrimento e configuración de servizos nun clúster. Entre as súas funcionalidades destacan o descubrimento de servizos (atopar novos provedores dun servizo dado), *health checking* (comprobar o estado de saúde dos nodos e servizos rexistrados) e un almacenamento xerárquico de tipo crave-valor. Na Figura 5 pódese observar a comunicación básica entre Consul, os procesos *master* e os procesos *worker*.

Actualmente é posible lanzar e destruír tarefas, consultar os *logs* das mesmas e consultar os recursos dispoñibles e estado das tarefas en Consul. Máis funcionalidades engadiranse no futuro como a pausa de tarefas e actualización de recursos de tarefas en execución. Ademais, unha topoloxía CATENAE pode ser despregada mediante ficheiros de definición escritos en linguaxe YAML.

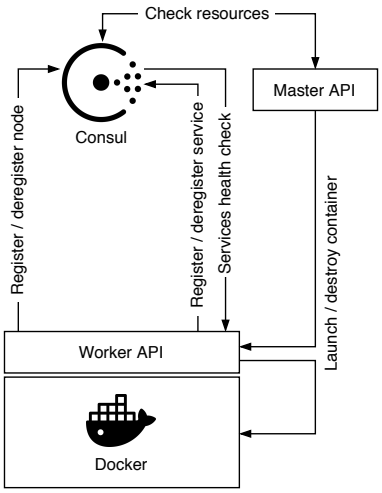

Figura 5: Comunicacións principais entre ANCORIS e Consul.

3. Software libre

3.1. Software libre empregado

Para a execución das topoloxías CATENAE emprégase Kafka e Docker, ambos con licenza Apache 2. Empréganse distintas librarías durante a execución de topoloxías, tamén con licenza Apache 2:

- Docker Client[7] como sistema de contedores para encapsular os micromódulos.
- Confluent Kafka[8] como cliente de Kafka.
- Aerospike Client[9].
- PyMongo[10] para o conector nativo para MongoDB.

No caso de REDD, para a topoloxía de clasificación empregouse Scikit-Learn[11] para a construción do clasificador e preprocesamento dos textos (BSD). Para a API HTTP empregouse Flask e Flask-RESTful (BSD). Na interfaz web:

- AngularJS[12] como *framework* base (MIT).
- MaterializeCSS[13] como base para o deseño e dinámicas básicas da web (MIT).
- JQuery[14] como dependencia de MaterializeCSS (MIT).
- NVD3[15] para as gráficas da evolución dos usuarios (Apache 2).

Ademais empregouse Python coas súas librarías básicas (licenza propia compatible con GPL).

[7] https://github.com/docker/cli
[8] https://github.com/confluentinc/confluent-kafka-python
[9] https://github.com/aerospike/aerospike-client-python
[10] https://github.com/mongodb/mongo-python-driver
[11] https://github.com/scikit-learn/scikit-learn
[12] https://github.com/angular/angular.js
[13] https://github.com/Dogfalo/materialize
[14] https://github.com/jquery/jquery
[15] https://github.com/novus/nvd3

3.2. Licenza utilizada

Tanto para CATENAE como para REDD e ANCORIS emprégase a licenza *GNU General Public License v3* (GPL-3) na súa última versión. Escóllese principalmente porque obriga a todo código derivado a licenzarse baixo a mesma licenza GPL (mantendo os produtos derivados cunha licenza libre) á vez que se permite o seu uso comercial. En resumo, a licenza permite un uso comercial do software, a creación de traballos derivados, a distribución do traballo orixinal e derivados e permite engadir garantías e reclamos de patentes por parte dos contribuidores. Pola contra, non permite utilizar licenzas distintas nin para o software orixinal nin para traballos derivados e descarga de responsabilidade ó propietario ou distribuidor por posibles danos. É obrigatorio incluir unha copia ou indicacións para atopar o software orixinal, indicar os cambios significativos realizados en traballos derivados, incluír o texto completo da licenza e manter o *copyright* orixinal. Tamén é preciso incluír instrucións de instalación nos traballos derivados.

3.3. Activade pública

Para os tres proxectos o código está dispoñible en GitHub. A través da plataforma xestiónanse as tarefas por resolver, a xestión de cambios e distribución de versións, e as incidencias dos usuarios das ferramentas. No caso da plataforma REDD, trátase dun prototipo final que resolve un caso de uso concreto. Sirve, non obstante, como un exemplo moi completo de referencia para implementar un sistema de procesamento de textos en tempo real con CATENAE; dende as topoloxías de procesamento até o almacenamento, consulta de información a través dunha API HTTP e a representación e interacción cos resultados a través nunha interfaz web. A través de Docker Hub distribúense as imaxes Docker dos micromódulos das topoloxías, así como da API e a interfaz web.

Un traballo derivado dunha das topoloxías, o *crawler de Reddit*[16], mantense actualizado de forma continuada e estase comezando a empregar nunha aplicación que permite escoller aqueles usuarios (sen extraer o seu historial) que *a priori* teñan maior probabilidade de ter unha alta puntuación nun clasificador particular.

O xestor de recursos ANCORIS está nunha etapa de desenvolvemento temperá. O obxectivo a curto prazo é a integración con CATENAE, permitindo despregues de topoloxías de forma transparente ó usuario e sen a necesidade de instalar ANCORIS, xa que execútase tamén en contedores Docker e pode despregarse inmediatamente antes da topoloxía, cunha vida útil ligada á mesma.

O proxecto máis maduro e activo é CATENAE, empregado actualmente noutros sistemas de recuperación de información en tempo real e incluso nun sistema *blockchain*. En GitHub conta cun desenrolo constante, engadindo novas funcionalidades e manténdose de forma continuada. A libraría distribúese empaquetada a través do popular repositorio de Python PyPi[17], permitindo a súa intalación de xeito sinxelo. Tamén se distribúe a través do Docker Hub unha imaxe base con todas as dependencias necesarias para empregar CATENAE (principalmente o cliente de Kafka) e os seus conectores (Aerospike, MongoDB). Ademais distribúese unha imaxe propia para executar un contedor Kafka[18], útil durante o desenrolo de topoloxías.

Por último, tamén a través de GitHub publícase a documentación dos proxectos en formato web grazas á ferramenta GitHub Pages. Todos os proxectos están abertos á contribucións a través de *pull requests*.

4. Conclusións

CATENAE é unha libraría Python que se desenvolve para dar soporte a un caso de uso particular desde unha perspectiva moito máis ampla, buscando a súa reutilización noutros proxectos de calquera índole con necesidades de escalabilidad e procesamiento en tempo real. Esta libraría é particularmente interesante no campo da Ciencia de Datos, permitindo a investigadores escalar as súas probas de concepto sen desperdiciar unha parte importante do proceso en cuestións tecnolóxicas tanxenciais dada a súa sinxeleza. Para dar soporte a esas necesidades de escalabilidade e seguindo a mesma filosofía, desenvólvese ANCORIS, un xestor de recursos para clústers con definición granular dos recursos e que permite executar contedores Docker.

A plataforma REDD para a detección temperá de depresión en Reddit constitúe un marco de traballo completo e totalmente operativo no que expertos na área poden validar ou invalidar as alertas emitidas polo sistema. O estado da arte en detección de depresión non permite un proceso totalmente automatizado. Por unha banda, a etiquetaxe de alertas como correctas e incorrectas (falsos positivos) permite mellorar o clasificador de signos de depresión. Dada

[16] https://github.com/brunneis/reddit-crawler
[17] https://pypi.org/project/catenae/
[18] https://hub.docker.com/r/catenae/kafka/

a magnitude de Reddit, é inviable analizar a todos os usuarios que participan nas distintas comunidades. Por iso, as alertas tamén funcionan como filtro, sendo posible alzar a voz de alerta ante os casos máis serios en tempo real. O sistema céntrase, como se comentou, na detección temperá. Así, outórgase especial relevancia á rapidez, aínda comprometendo a precisión do sistema a través da redución do umbral mínimo de probabilidade para desencadear unha alerta. Ademais da xestión de alertas, é posible realizar a descarga dos conxuntos de datos xerados a través da etiquetaxe e consultar estatísticas sobre o desempeño do sistema en tempo real.

5. Anexo

Unha demostración en vídeo da plataforma REDD executada sobre ANCORIS está dispoñible en YouTube[19].

Referencias

[1] D. Losada, F. Crestani, and J. Parapar, "eRISK 2017: CLEF Lab on Early Risk Prediction on the Internet: Experimental Foundations," in *Proc. of CLEF*, pp. 346–360, 2017.

[2] Docker. http://www.docker.com/, 2018. [Online; accessed September, 2018].

[3] R. Martínez-Castaño, J. C. Pichel, D. E. Losada, and F. Crestani, "A Micromodule Approach for Building Real-Time Systems with Python-Based Models: Application to Early Risk Detection of Depression on Social Media," in *Advances in Information Retrieval*, pp. 801–805, Springer, 2018.

[4] R. Martínez-Castaño, J. C. Pichel, D. E. Losada, and F. Crestani, "Building python-based topologies for massive processing of social media data in real time," in *5th Spanish Conference in Information Retrieval*, ACM, 2018.

[5] Scikit-Learn. http://scikit-learn.org/, 2018. [Online; accessed September, 2018].

[6] Apache Storm. https://storm.apache.org/, 2018. [Online; accessed September, 2018].

[7] Apache Kafka. https://kafka.apache.org/, 2018. [Online; accessed September, 2018].

[8] Aerospike. https://www.aerospike.com/, 2018. [Online; accessed September, 2018].

[9] D. Losada and F. Crestani, "A Test Collection for Research on Depression and Language Use," in *Proc. of CLEF*, pp. 28–39, 2016.

[10] Consul. https://www.consul.io/, 2018. [Online; accessed September, 2018].

[19] https://www.youtube.com/watch?v=52tZNJV5d1E