

UNIVERSIDAD DE CHILE

FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

**TEXT MINING APLICADO A DOCUMENTACIÓN DE API PARA LA DETECCIÓN DE DIRECTIVAS**

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

GABRIEL CORREA GAETE

PROFESOR GUÍA:

ROMAIN ROBBES

MIEMBROS DE LA COMISIÓN:

JORGE PÉREZ R.

BÁRBARA POBLETE L.

SANTIAGO DE CHILE

2015

Índice

[1 Introducción 1](#_Toc425868280)

[1.1 Documentación de API 1](#_Toc425868281)

[1.2 Problema y Motivación 1](#_Toc425868282)

[1.3 Una Solución 1](#_Toc425868283)

[1.4 Objetivos 1](#_Toc425868284)

[1.4.1 Objetivos Generales 1](#_Toc425868285)

[1.4.2 Objetivos Específicos 1](#_Toc425868286)

[1.4.3 Metodología 1](#_Toc425868287)

[2 Marco teórico y trabajos anteriores 1](#_Toc425868288)

[2.1 Importancia de comentarios en APIs y código fuente 1](#_Toc425868289)

[2.2 Destacación de directivas en APIs 1](#_Toc425868290)

[2.2.1 Problema: Las directivas pueden pasar desapercibidas 1](#_Toc425868291)

[2.2.2 eMoose 1](#_Toc425868292)

[2.2.3 Syntax highlighting en editores de texto: Sublime Text 1](#_Toc425868293)

[2.3 Trabajo de Monperrus et al. 2](#_Toc425868294)

[2.3.1 Keywords para detectar directivas 2](#_Toc425868295)

[2.3.2 Taxonomía de directivas 2](#_Toc425868296)

[3 Trabajo Realizado: Text Mining sobre comentarios de API 2](#_Toc425868297)

[3.1 Preparación de los datos 2](#_Toc425868298)

[3.1.1 Datos del trabajo de Monperrus 2](#_Toc425868299)

[3.1.2 Datos necesarios para aplicar Text Mining 2](#_Toc425868300)

[3.1.3 Proceso de adaptación de los datos 2](#_Toc425868301)

[3.2 Implementación de aplicación para la revisión de comentarios 2](#_Toc425868302)

[3.3 Diseño del proceso de Text Mining 3](#_Toc425868303)

[3.3.1 Text Mining en Weka 3](#_Toc425868304)

[3.3.2 Semántica de los datos 3](#_Toc425868305)

[3.3.3 Clasificadores 3](#_Toc425868306)

[3.3.4 Experimentos realizados usando Weka 3](#_Toc425868307)

[3.4 API de Weka 3](#_Toc425868308)

[4 Resultados 3](#_Toc425868309)

[4.1 Resultados de los Experimentos realizados en Weka 4](#_Toc425868310)

[4.1.1 Tablas comparando distintos clasificadores, usando o no stemming, usando o no stopwords 4](#_Toc425868311)

[4.1.2 Tablas adicionales comparando otras opciones y parámetros del filtro y opciones del mejor clasificador 4](#_Toc425868312)

[4.2 Discusión de los resultados 4](#_Toc425868313)

[5 Limitaciones y problemas de validez 4](#_Toc425868314)

[6 Conclusiones y trabajo futuro 4](#_Toc425868315)

[7 Glosario 4](#_Toc425868316)

[8 Bibliografía 4](#_Toc425868317)

[9 Anexo 4](#_Toc425868318)

# Introducción

## Documentación de API

Explicar que es la documentación de api y más.

## Problema y Motivación

Directivas no se ven. Esto hace mas probable la existencia de bugs. Intentar escribir mejor documentación suena como una solución, pero puede haber varios problemas para que funcione en la practiva [[falta una referencia a algún estudio para la ultima frase]]

Una directiva es … [[definición corta aca]]. .. [[Para una explicación detallada del concepto ver [°n] o Anexo/Directivas]]

## Una Solución

[[Hablar de destacar directivas y de la necesidad de mayor automatización pues la asignación manual de directivas es muy tedioso y toma mucho tiempo.]]

Una solución – la propuesta por este trabajo – es que se detecten de manera automática las directivas de API usando las herramientas de Text Mining y Machine Learning, usando comentarios reales de librerías API como base y aprendizaje.

Es posible que un programa aprenda a detectar directivas, la duda es ¿con cuánta eficacia o precisión lo hace? El rendimiento de este método es medido en el presente trabajo en una serie de experimentos usando Text Mining, y es comparado con el rendimiento de otro método: si un comentario contiene alguna palabras clave, o *keyword*, de cierto conjunto preestablecido, entonces es una directiva.

La solución incluye un software que facilita el pre-procesamiento de los comentarios de API necesario para la aplicación de Text Mining. Permite realizar una asignación de directiva/no-directiva (y posiblemente más tipos) de cada phrase de los comentarios de una API y luego entregar los resultados en un formato compatible con la aplicación de Text Mining.

## Objetivos

(parte similar al informe de CC6908)

### Objetivos Generales

### Objetivos Específicos

### Metodología

# Marco teórico y trabajos anteriores

## Importancia de comentarios en APIs y código fuente

* % de comentarios en código fuente
* Trabajos pasados que apoyan la inclusión de comentarios para ayudar a desarrolladores

## Destacación de directivas en APIs

### Problema: Las directivas pueden pasar desapercibidas

* Mostrar ejemplos de directivas no vistas

### Syntax highlighting en editores de texto

* Mencionar que es posible extender eclipse y sublime text para que haga highlight de ciertas cosas.. que esto podría ayudar a destacar directivas adentro de los editores.
* Mostrar imágenes de la prueba realizada en sublime text

### eMoose y tagSea

* Resultados de este estudio con las directivas destacadas en Eclipse
* Imágenes de eMoose
* Explicación de eMoose y su posible utilidad y poder
* Taxonomia propuesta

### Sublime Text

* Hablar de que se probó que en sublime text es posible destacar frases o palabras claves (mostrar 2 imágenes de ejemplo) (mencionar esto pues puede ser otro camino de solución al destacar directivas)

## Trabajo de Monperrus et al.

Mencionar que el trabajo presenta una buena referencia de comparación al detectar directivas automáticamente, pues el trabajo obtuvo los porcentajes de keywords en cada directiva que encontraron

### Keywords para detectar directivas

### Taxonomía de directivas

# Desarrollo de <NombrePrograma>

Nombres posibles: Comments Highlighter (CHI o CHi); Java Api Highlighter (JAH); Jah Api Highlighter (JAH)

* Extracción de comentarios de varios archivos en una carpeta
* Objetivo y propósito de la Aplicación (Y utilidad .incluuir también.que se necesita paraTextMining)
* Pagina Web. Xq HTML5- > POR compatibilidad.
* Mostrar capacidades de la página, imágenes, input, output, hablar de importancia de hotkeys, hablar de dificultades encontradas para lograr una buena visualización de comentarios
* Visualización de comentarios
* Extraccion de comentarios
* Pre-proceso de comentarios
* Exportacion de resultados
* Desarrollo Futuro e Ideas Propuestas

# Revisión Manual de Comentarios de API

* Por qué una revisión manual (por Weka .arff, por datos de Monperrus balanceados dir vs non-dir, por necesidad de datos comprendidos por nosotros y para usarlos en Weka)
* Explicar Datos del trabajo de Monperrus. Mencionar que los datos son directos de las APIs java, apache.commons y eclipse.jface
* Preparacion de los datos de Monperrus, venían en .xml. los necesitamos en .csv
* Conjunto de datos preliminar (mencionar que sirvió de prueba y entregó información, intuición)
* Conjunto de datos final. 500 comentarios de 3 apis. Randomizado. Sin @throws. Solo Metodos. Solo mayores a 3 o 4 (revisar cuantas) líneas.
* Método de Revisión (una explicación de cómo fueron revisados los comentarios)(mencionar a los revisadores: yo y Romain)
* Discusión y Observaciones

Fue necesario realizar una revisión manual de comentarios para obtener datos requeridos por las herramientas de Machine Learning. Es decir, para que el programa tuviera ejemplos con los cuales pudiera aprender a clasificar comentarios de API.

Cada comentario de API fue separado por las frases que lo componen, y cada frase fue asignada una de las siguientes clases: no-directiva, directiva, consejo, directiva-null.

La revisión manual de comentarios fue realizada usando <NombrePrograma>. El uso de la herramienta sin duda agilizó el proceso de revisión principalmente porque mejora la legibilidad de los comentarios y porque se puede usar hotkeys (o teclas de acceso rápido) para navegar por la lista de comentarios y asignar clases a las frases.

Se realizaron dos sesiones de revisión manual, la con los datos preliminares y la con los datos finales. Los datos preliminares revelaron algunas fallas de diseño y validez explicados más adelante y ayudaron a entregar información de cómo diseñar los datos finales. Ambas son descritas a continuación.

### Revisión de los datos preliminares

Los datos preliminares consisten de 1894 comentarios javadoc formando un total de 8876 frases. Los comentarios javadoc usados consisten cada uno de un bloque de comentario multilinea correspondiente a un método, campo o clase como en el ejemplo a continuación:

/\*\*

\* Constructs a new empty <code>ArrayStack</code>. The initial size

\* is controlled by <code>ArrayList</code> and is currently 10.

\*/

Los comentarios corresponden a un subconjunto de la documentación de Eclipse JFace. [[last]]

### Revisión de los datos finales

Cada frase de los comentarios fue leída y asignada una clase por el autor de este trabajo y por e

# Aplicación de Text Mining

* Sobre Weka . Introduccion. Interfaz . existencia de una api. (Mejor incluir sección de Weka explicándolo)
* Formato de Entrada para hacer text Mining en Weka. Unigramas.
* (explicar en alguna parte le tranformacion de .csv del output de la pagina a .arff)
* Filtros, clasificadores, selección de atributos. (Mejor incluir sección de Weka explicándolo)
* Describir los clasificadores usados. Como. Con qué parámetros? Detalles. Mostrar resultados en la próxima sección
* Describir las combinaciones de stemming, stopwords y parámetros usados
* Mencionar cuales son los que pueden ser usados como machine learning ., aperndiendo con cada set de datos nuevos, usando la api de java.
* Mencionar overfitting y lo que se hizo para evitar tener overfitting

# Resultados

## Resultados Preliminares

* Hablar de los resultados con el primer set de datos (verificar que ya mencioné estos datos y esta sección en 3(Trabajo Realizado) (incluir una referencia a cuales son los datos preliminares, para que no haya confusión)
* Tablas comparando distintos clasificadores, usando o no stemming, usando o no stopwords
* Tablas adicionales comparando otras opciones y parámetros del filtro y opciones del mejor clasificador

## Resultados Finales

* Hablar de los resultados con el set de datos final (incluir una referencia a cuales son los datos finales, para que no haya confusión) Discusión de los resultados
* Tablas comparando distintos clasificadores, usando o no stemming, usando o no stopwords
* Tablas adicionales comparando otras opciones y parámetros del filtro y opciones del mejor clasificador

# Discusión general

* Mencionar en alguna parte lo de sublime text (y quizás eMoose)

# Limitaciones y problemas de validez(?)

# Conclusiones y trabajo futuro

# Glosario

# Bibliografía

# Anexo