Sistema de Detección de Trending Topics usando Count-Min Sketch y Ventanas Deslizantes para Wordcloud en Tiempo Real

RODRIGO SILVA, Universidad, País

Este artículo presenta un sistema eficiente para detección de trending topics en tiempo real utilizando estructuras de datos probabilísticas. El enfoque combina Count-Min Sketch para conteo aproximado, ventanas deslizantes para análisis temporal y Min-Heap para mantenimiento eficiente del top-K. El sistema genera wordclouds dinámicos por ventana temporal, permitiendo visualizar la evolución de temas relevantes. Se demuestra el proceso completo con ejemplos detallados desde el preprocesamiento de texto hasta la generación del wordcloud final.

Additional Key Words and Phrases: Count-Min Sketch, Trending Topics, Wordcloud, Tiempo Real, Procesamiento de Texto

ACM Reference Format:
Rodrigo Silva. 2024. Sistema de Detección de Trending Topics usando Count-Min Sketch y Ventanas Deslizantes para Wordcloud en Tiempo Real. *Proc. ACM Meas. Anal. Comput. Syst.* 41, 3, Article 111 (December 2024), 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introducción

La detección de trending topics en flujos continuos de texto representa un desafío computacional significativo. Plataformas como Twitter y Facebook procesan millones de publicaciones diarias, requiriendo algoritmos que combinen eficiencia en memoria y velocidad de procesamiento. Este trabajo propone un sistema basado en Count-Min Sketch para identificar temas trending en ventanas temporales deslizantes, generando wordclouds que reflejan la relevancia temporal de los temas.

La principal contribución es un pipeline completo que incluye preprocesamiento de texto, conteo probabilístico eficiente y generación de visualizaciones dinámicas, todo ello con complejidad de memoria constante independiente del volumen de datos.

2 Arquitectura del Sistema

2.1 Componentes Principales

El sistema consta de tres componentes fundamentales:

- **Preprocesador de Texto**: Estandarización, limpieza y tokenización
- **Count-Min Sketch**: Conteo aproximado de frecuencias
- **Min-Heap**: Mantenimiento eficiente del top-K

2.2 Flujo de Procesamiento

El procesamiento sigue una arquitectura de ventana deslizante donde:

(1) Se procesan lotes de textos en ventanas de tamaño fijo k

Author's Contact Information: Rodrigo Silva, Universidad, Ciudad, País.

ACM 2476-1249/2024/12-ART111
https://doi.org/XXXXXXX.XXXXXXX

(2) Cada ventana genera un wordcloud independiente
(3) Al deslizar la ventana, se descartan los textos más antiguos
(4) Solo se mantiene estado de la ventana actual

3 Metodología Detallada

3.1 Preprocesamiento de Texto

Cada texto sigue un pipeline de transformación:

Ejemplo con texto de entrada:

```
"El partido de fútbol estuvo INCREÍBLE! #Deporte"
```

Pasos de preprocesamiento:

(1) **Minúsculas**: "el partido de fútbol estuvo increíble! #deporte"
(2) **Limpieza**: "partido de fútbol estuvo increíble deporte"
(3) **Stopwords**: "partido fútbol increíble deporte"
(4) **Tokens finales**: ["partido", "fútbol", "increíble", "deporte"]

3.2 Count-Min Sketch en Acción

El Count-Min Sketch utiliza una matriz de $depth \times width$ para conteo aproximado. Para un sketch de 3×5:

Inserción de "fútbol":

- Hash1("fútbol") = 2 → Incrementar posición [0,2]
- Hash2("fútbol") = 4 → Incrementar posición [1,4]
- Hash3("fútbol") = 1 → Incrementar posición [2,1]

Estado del Sketch después de múltiples inserciones:

```
    Col0 Col1 Col2 Col3 Col4
Fila0 [ 0 , 0 , 3 , 0 , 1 ]
Fila1 [ 0 , 1 , 1 , 0 , 2 ]
Fila2 [ 1 , 2 , 0 , 0 , 1 ]
```

Consulta de "fútbol":

- Hash1("fútbol") = 2 → Valor = 3
- Hash2("fútbol") = 4 → Valor = 2
- Hash3("fútbol") = 1 → Valor = 2
- Frecuencia estimada = min(3, 2, 2) = 2

3.3 Mantenimiento del Top-K con Min-Heap

El Min-Heap de tamaño k mantiene eficientemente los elementos más frecuentes:

Ejemplo con k=3:

(1) Heap inicial vacío: []
(2) Insertar "fútbol" (freq=4): [(4, "fútbol")]
(3) Insertar "partido" (freq=1): [(1, "partido"), (4, "fútbol")]
(4) Insertar "increíble" (freq=1): [(1, "partido"), (4, "fútbol"), (1, "increíble")]
(5) Llega "gol" (freq=2): 2 > mínimo(1) → extraer mínimo, insertar nuevo
(6) Heap final: [(2, "gol"), (4, "fútbol"), (1, "increíble")]

4 Experimento y Resultados

4.1 Configuración del Experimento

Se procesaron 8 textos con los siguientes parámetros:

- Tamaño de ventana: $k = 3$ textos
- Top-K: 3 trending topics por ventana
- Count-Min Sketch: 3 filas × 1000 columnas

4.2 Procesamiento por Ventanas

Ventana 1 - Textos [1,2,3]:

- **Textos procesados**: 3 textos deportivos
- **Top-3 estimado**: ["fútbol", "gol", "partido"]
- **Frecuencias**: fútbol=4, gol=2, partido=1
- **Wordcloud**: **FÚTBOL** gol partido

Ventana 2 - Textos [2,3,4]:

- **Textos procesados**: 2 deportivos + 1 sobre terremoto
- **Top-3 estimado**: ["terremoto", "fútbol", "gol"]
- **Frecuencias**: terremoto=4, fútbol=3, gol=1
- **Wordcloud**: **TERREMOTO** fútbol gol

Ventana 3 - Textos [3,4,5]:

- **Textos procesados**: 1 deportivo + 2 sobre terremoto
- **Top-3 estimado**: ["terremoto", "alerta", "ciencia"]
- **Frecuencias**: terremoto=5, alerta=1, ciencia=1
- **Wordcloud**: **TERREMOTO** alerta ciencia

4.3 Análisis de Resultados

El sistema demostró capacidad para:

- **Detectar cambios rápidos**: Transición de temas deportivos a emergencia (terremoto)
- **Mantenimiento eficiente**: Uso constante de memoria independiente del número de palabras únicas
- **Respuesta en tiempo real**: Procesamiento incremental por ventanas

Evolución de Trending Topics:

```
Ventana 1: [fútbol, gol, partido]
Ventana 2: [terremoto, fútbol, gol]     ← Detección de evento
Ventana 3: [terremoto, alerta, ciencia] ← Consolidación
```

5 Discusión

5.1 Ventajas del Enfoque

- **Eficiencia en memoria**: Count-Min Sketch usa espacio constante
- **Escalabilidad**: Rendimiento independiente del volumen de datos
- **Actualizaciones incrementales**: Procesamiento por ventanas
- **Tolerancia a colisiones**: Múltiples funciones hash minimizan errores

5.2 Limitaciones

- **Conteo aproximado**: Posible sobre-estimación por colisiones

- **Pérdida de histórico**: Solo se mantiene ventana actual
- **Sensibilidad a parámetros**: Tamaño de sketch y número de hashes

6 Conclusión

El sistema presentado provee una solución eficiente y escalable para detección de trending topics en tiempo real. La combinación de Count-Min Sketch y ventanas deslizantes permite generar wordclouds dinámicos que reflejan accuratemente la evolución temporal de temas relevantes. El enfoque es particularmente adecuado para aplicaciones que requieren procesamiento de flujos continuos de texto con recursos limitados.

El trabajo futuro incluye la incorporación de detección de burst scores usando EWMA y la extensión a múltiples ventanas temporales simultáneas para capturar patrones a diferentes escalas de tiempo.

Acknowledgments

Este trabajo fue desarrollado como parte de una investigación sobre estructuras de datos probabilísticas para procesamiento de lenguaje natural en tiempo real.