1. Desarrollo de un prototipo de aplicación web para extraer ofertas laborales (Web Scraping) en la Región Caribe

Objetivo: Construir un prototipo de aplicación web que extraiga ofertas laborales de las principales plataformas de empleo de la Región Caribe mediante **web scraping**.

a) Marco legal del web scraping en Colombia y LATAM (¿es legal?)
Presentar el panorama regulatorio y las consideraciones de cumplimiento (términos de uso, protección de datos personales, derechos de autor, medidas técnicas de acceso, etc.), con ejemplos y referencias por país.

b) Contexto técnico de web scraping (para referencias bibliográficas)

1. Conceptos clave: crawling vs. scraping, parsers HTML, extracción estructurada.
2. Módulos y herramientas comunes: requests/HTTP, manejo de sesiones y cookies, rotación de *user-agents* y proxys, control de *rate limiting*, parsing (p. ej., BeautifulSoup, lxml), *headless browsers* (p. ej., Playwright/Puppeteer), y *pipelines* de limpieza y almacenamiento.
3. Buenas prácticas: respeto de `robots.txt`, *backoff* exponencial, *throttling*, validación y normalización de datos, registro de fuentes y sellos de tiempo, *unit tests* para selectores, y monitoreo de cambios en el DOM.

2. Demanda de habilidades tecnológicas: evidencia desde el mercado laboral colombiano

a) Extracción y limpieza de datos
Se construyó una base con ofertas y vacantes publicadas entre **enero de 2018** y **diciembre de 2023**, utilizando información de **elempleo.com**, una de las principales bolsas de empleo virtual del país.
Durante la recolección se detectaron ofertas con descripciones incompletas o con variables clave ausentes (p. ej., nivel mínimo de estudios, años de experiencia, sector).
El proceso de limpieza incluyó:

- **Eliminación** de registros con valores nulos críticos.
- **Corrección** de inconsistencias mediante dos enfoques complementarios:
    - **Diagnóstico estadístico:** detección de valores atípicos en variables como tiempo entre publicación y cierre, número de vacantes, etc.
    - **Diagnóstico conceptual/temático:** verificación de completitud de títulos y descripciones, y congruencia entre nivel educativo, profesión y años de experiencia requeridos.

b) Definición operativa de "habilidades tecnológicas"
Conjunto de capacidades que permiten a una persona interactuar con TIC en distintos niveles de complejidad: desde el uso especializado de software y gestión de redes, hasta el acceso a

información, trabajo remoto y comunicación/promoción en plataformas digitales.
se plantea una **tipología** con cuatro dimensiones:

1. Teletrabajo
2. Habilidades tecnológicas generales
3. Habilidades tecnológicas especializadas
4. Habilidades de Tecnologías de la Información y las Comunicaciones (TIC)

c) Marco teórico (actividades, tareas y automatización)
La publicación de vacantes en línea ha reconfigurado el mercado laboral reciente al mejorar el emparejamiento entre empleadores y trabajadores. Evidencia para Colombia sugiere que una mayor penetración de vacantes *online* se asocia con menor tasa de vacantes para un mismo nivel de desempleo, reduciendo fricciones en el mercado. Asimismo, se documenta que la pandemia aumentó el riesgo de automatización, afectando más a trabajadores en sectores informales y de baja calificación. En economías con baja inversión tecnológica, choques externos que encarecen el capital humano pueden acelerar la automatización.
Vacío identificado: falta explorar cómo ajustarán las empresas sus decisiones en el mediano plazo pasado el retorno al empleo prepandémico. Este documento aporta una tipología y un análisis de un periodo amplio posterior a la crisis, útil como base para trabajos futuros.

d) Uso de O*NET como referencia
Para detectar habilidades tecnológicas **generales**, se toma como referencia un conjunto de habilidades de alta demanda reportadas por **O*NET**, portal de empleo financiado por el Departamento de Trabajo de EE. UU.

e) Tipología y método de clasificación
Se distinguen **cuatro** tipos de habilidades tecnológicas:

1. **Generales**
2. **Especializadas**
3. **TIC**
4. **Teletrabajo**

La asignación a categorías se realiza a partir del **título**, la **descripción** y la **categorización sectorial** de cada oferta, mediante un **algoritmo de emparejamiento de texto**.

f) Criterios para identificar si una vacante es "tecnológica"
i. **Agrupación por tipo de habilidad**

- **Habilidades especializadas:** necesarias para producir bienes/servicios TIC (p. ej., SQL, Python, Java, SAP) y para programar, desarrollar aplicaciones o gestionar redes.
- **Habilidades genéricas:** de uso transversal en múltiples ocupaciones (p. ej., Excel, Word, Outlook, Google Workspace), acceso a información en línea y reuniones virtuales.
- **Habilidades tecnológicas complementarias:** comunicación en redes sociales y posicionamiento en plataformas de comercio electrónico (p. ej., Instagram, Facebook, Teams, Slack).

ii. **Variable dicotómica de especialización**
Se crea una variable binaria para identificar ofertas que requieren **habilidades especializadas** típicas del sector de **Tecnología, Información y Comunicaciones**, asociadas a uso intensivo de conocimientos digitales.

iii. **Regla sectorial y de palabras clave**
Si la vacante pertenece al **sector de Sistemas y Tecnología** y el **título/descrición** contiene términos del dominio (p. ej., "TIC", "tecnologías de la información", "sistemas"), se clasifica como demanda de habilidades tecnológicas.

iv. **Vinculación con ocupaciones (CIUO)**
Se relacionan tareas descritas por las firmas con las de la **Clasificación Internacional Uniforme de Ocupaciones (CIUO)** mediante emparejamiento de texto entre **descripción de la vacante ↔ descripción de la ocupación** y **título de la vacante ↔ título de la ocupación**.

- Se considera **clasificación exitosa** si la similitud supera **80 %**.
- Con este método se clasificó **52 %** de las vacantes a **4 dígitos** de la CIUO.
- Para el **48 %** restante, se aplicó clasificación a **2 dígitos**, logrando cubrir **90 %** de las vacantes.
- Un **5 %** adicional se clasificó con **revisiones manuales** e imputación apoyada en otras variables.

Nota: no se adopta CIUO como base principal para **habilidades** porque ofrece menor granularidad específica frente a **ESCO**; por ello, para *skills* se prioriza una taxonomía más rica (p. ej., ESCO) y CIUO se utiliza como referencia ocupacional.

Que no me gusta? Usó un método clásico de n-gramas + similitud de cadenas **(con tokenización posterior) para calcular el parecido entre textos. Faltan spacy, ner, embeddings, algo mas moderno.**

3. Enhancing Skills Demand Understanding through Job Ad Segmentation Using NLP and Clustering Techniques

a) Que hace (SUUUUPER alineado a nosotros, nos sirve un monton, la justificacion tambien)
El artículo aborda la transformación del mercado laboral bajo presiones de globalización, digitalización, cambios demográficos y automatización. El foco está en Lituania, pero la metodología es extrapolable a otros países. La hipótesis central es que el uso de NLP (Natural Language Processing) combinado con técnicas de clustering permite evaluar en tiempo real los cambios en la demanda de habilidades, superando las limitaciones de métodos tradicionales como encuestas periódicas

Extracción de datos (Web Scraping y preprocesamiento)

- **Fuente de datos**: +500.000 ofertas laborales de portales lituanos.

- **Tecnologías de scraping**: Python con **Playwright, Selenium y BeautifulSoup** para capturar datos no estructurados.
- **Problema central**: las ofertas varían en estructura; las secciones "Requisitos" o "Competencias" no siempre aparecen ni son consistentes en formato.
- **Almacenamiento**: se usó **PostgreSQL** para permitir análisis paralelo y consultas complejas

Para extraccion:
El artículo contrasta varios enfoques:

- **Búsqueda de palabras clave y reglas (POS patterns)**: rápido pero dependiente de diccionarios estáticos.
- **Expresiones regulares (regex)**: resultaron ser la estrategia más robusta para aislar secciones como *Requirements → Company offers*.
- **LLMs (GPT-3.5 y GPT-4)**: se probaron para extracción directa de requisitos.
    - **Ventaja**: mayor semántica, resultados más precisos.
    - **Desventaja**: costos prohibitivos (320M tokens requeridos).
- **NER (Named Entity Recognition)**: identificada como la técnica más prometedora a futuro, pero no implementada por falta de dataset etiquetado

Para transformar los textos en datos numéricos, se evaluaron distintas técnicas:

- **TF-IDF**: eficiente, pero limitado en semántica.
- **Sentence Transformers (BERT)**: adoptados como método principal (384 dimensiones), capturando significado contextual.
- Métodos alternativos discutidos: **Word2Vec, GloVe, FastText, Doc2Vec, BoW**.
- **Trade-off**: modelos tipo BERT ofrecen embeddings de alta calidad, pero con mayor complejidad y costo computacional

Los embeddings generan vectores muy grandes (Nx384). Se aplicaron técnicas para reducir la dimensionalidad antes del clustering:

- **PCA (lineal)**
- **t-SNE (no lineal, fuerte en visualización local)**
- **UMAP (no lineal, escalable, preserva estructura local y global)**
- **Trimap e Isomap** como alternativas.

Resultado: UMAP ofreció el mejor desempeño (según la métrica de *trustworthiness*), seguido de t-SNE

El artículo evalúa una batería amplia de métodos:

- **K-means**: eficiente pero limitado a clusters esféricos.
- **DBSCAN / HDBSCAN**: robustos frente a ruido, capturan clusters de densidad variable. HDBSCAN mostró mejor estabilidad.
- **BIRCH**: escalable a grandes volúmenes, útil para documentos largos.

- **Affinity Propagation**: no requiere número de clusters, pero costoso.
- **Spectral Clustering**: captura relaciones no lineales, aunque caro computacionalmente.
- **CBMIDE y RCBMIDE**: métodos novedosos de densidad basados en fórmulas de inversión modificada, mostraron rendimiento competitivo.

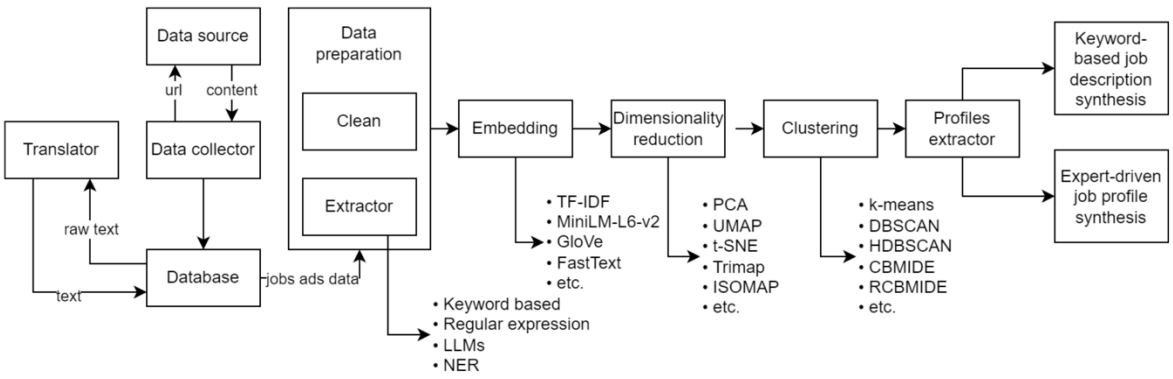
Resultado global: HDBSCAN + UMAP fue la combinación más robusta

- **Pipeline híbrido:** regex (extracción inicial) → BERT embeddings → UMAP → HDBSCAN → generación de perfiles.

- **Uso de LLMs**: probado como alternativa, descartado por costos, pero proyectado como línea futura.

IMPORTANTE: Cogen el job ad, tratan de buscar donde esta la seccion de requerimientos, extraen esa seccion, y le hacen embeddings a esa seccion. Nosotros proponemos, al estar enfocados en la extraccion de habilidades especificas, extraer cada habilidad individual, coger el corpus de habilidades requeridas, hacerles embeddings a las habilidades, luego cluster, para identificar asi perfiles o grupos de habilidades para enfocar mejor decisiones gubernamentales, laborales o educativas. Aparte sirve dado que nos van a salir habilidades que no estan en esco so complementamos.

Extraen keywords para mapear demanda de lenguajes de programacion



---

4. Skill-LLM: Repurposing General-Purpose LLMs for Skill Extraction

a) Qué hace

El artículo propone finetunear LLMs generales (ej. LLaMA 3 8B) para la **extracción de habilidades en descripciones de empleo**, combinando supervisión tradicional con el poder de modelos masivos. Introducen **Skill-LLM**, un modelo fine-tuneado en el dataset **SkillSpan**, y comparan su desempeño frente a métodos clásicos (BERT/SpanBERT, ESCOXLM-R, JobBERT) y enfoques recientes con LLMs prompt-based (GPT-3.5/4, NNOSE).
También presentan una alternativa ligera: **GLiNER** fine-tuneado, útil en escenarios con limitaciones de cómputo. MALO: Skillspan esta tagged en ingles, y el finetuned model tambien esta en ingles. No hay datasets requeridos en espanol para hacer proper finetuning.

Justificación:

- Los enfoques previos basados en diccionarios, n-gramas y NER clásicos fallan al capturar la variabilidad semántica de las ofertas.
- Los LLMs en modo zero-shot/few-shot mejoran flexibilidad, pero generan salidas inconsistentes, poco parsables y con "alucinaciones". (VIEJO)
- La hipótesis es que **el fine-tuning de un LLM generalista permite superar esas limitaciones**, manteniendo parsabilidad y mejorando el F1.

b) Extracción y metodología

1. **Formato de salida**:
   - Diseñan prompts para que el modelo devuelva resultados en JSON con `skill_span` + `context`, asegurando que cada entidad pueda mapearse al texto original.
   - Ejemplo: `"QA"` como conocimiento con contexto `"all QA relevant"`.
   - Esto resuelve problemas comunes de NER con múltiples apariciones de una misma palabra (ej. "office").
2. **Modelos probados**:
   - **Skill-LLM (LLaMA 3 8B + LoRA)** → enfoque principal.
   - **GLiNER** → modelo liviano, entrenado con etiquetas estilo NER.
3. **Dataset**:
   - **SkillSpan** (Zhang et al. 2022) → frases de 3 fuentes (tech, house, big).
   - Contiene spans anotados de *skills* y *knowledge*.
   - Ejemplo de distribución: ~2200 spans de skills en train, ~1100 en test
4. **Baseline comparados**:
   - JobBERT / SpanBERT (NER clásico).
   - ESCOXLM-R (preentrenado en ESCO multilingual).
   - GPT-3.5 / GPT-4 (zero/few-shot, extract-style y NER-style).
   - NNOSE (retrieval + nearest neighbors).
5. **Entrenamiento**:
   - LLaMA 3 8B con LoRA (rank=64).
   - Fine-tuning en 2 epochs, batch size=4, learning rate=2e-4.
   - Evaluación con **Span F1** (exact match de spans).

c) Resultados

- **Skill-LLM (LLaMA 3 8B fine-tuneado)** logra:
  - F1 total: **64.8%** (vs. 64.2% de NNOSE, 62.6% de ESCOXLM-R, 58.9% de JobSpanBERT).
  - F1 en *skills*: 54.3%
  - F1 en *knowledge*: 74.2%
- **GLiNER fine-tuneado (166M params)**:
  - F1 total: 58.4%, comparable a JobBERT/SpanBERT pese a ser mucho más pequeño.
- **LLMs prompt-based** (GPT-3.5/4 zero/few-shot):
  - Muy por debajo (17.8–27.8% F1).
  - Principales problemas: inconsistencia, outputs no parsables, alucinaciones.

d) Aportes clave

1. **Parsabilidad robusta**: su formato JSON con contexto evita el problema clásico de outputs caóticos en LLMs prompt-based.
2. **Mejora de SOTA**: Skill-LLM supera modelos especializados como ESCOXLM-R y JobBERT, sin necesidad de pretraining adicional en datos del dominio.
3. **Trade-off exacto**:
   - Modelos grandes → mayor precisión pero lentos y costosos.
   - Modelos livianos → menores recursos, resultados aceptables.
4. **Discusión crítica**:
   - NNOSE "trampea" al fusionar skills y knowledge en una sola clase, inflando métricas.
   - En cambio, Skill-LLM mantiene la distinción y aún así gana.
5. **Limitaciones**:
   - Requiere fine-tuning (no sirve directo en zero-shot).
   - Computacionalmente más caro que NER clásico.
   - Algunos errores en etiquetas de dataset impactan métricas (≈8% de casos).

IMPORTANTE: DADO QUE NO TENEMOS DATASETS ANOTADOS, NO SERIA FACILMENTE MEDIBLE UN F1 SCORE

5. SKILLSPAN: Hard and Soft Skill Extraction from English Job Postings

a) Qué hace

El artículo presenta **SKILLSPAN**, un dataset anotado a nivel de *span* para extraer tanto **hard skills** como **soft skills** en descripciones de empleo en inglés. Es el **primer corpus grande con guías de anotación públicas** y con spans anotados por expertos en dominio (no crowdworkers), lo que lo convierte en referencia obligada en la literatura.

Además, los autores evalúan modelos basados en BERT y variantes adaptadas al dominio (JobBERT y JobSpanBERT), así como configuraciones de *single-task learning* (STL) y *multi-*

task learning (MTL). La hipótesis central es que la **pre-entrenamiento continuo en textos de vacantes laborales** mejora sustancialmente el rendimiento de modelos de lenguaje en tareas de extracción de habilidades

b) Dataset y anotación

1. **Fuentes de datos (2012–2021)**:
    - **BIG**: Plataforma general con variedad de puestos.
    - **HOUSE**: Dataset institucional (agencia laboral danesa).
    - **TECH**: StackOverflow Jobs, enfocado en puestos técnicos.

    Total: **391 postings anotados**, 14.5K oraciones, 232K tokens, 12.5K spans de habilidades/conocimientos
    Además, incluyen **126.8K postings sin anotar** (3.2M oraciones, 460M tokens) para preentrenamiento continuo.

2. **Proceso de anotación**:
    - Duró 8 meses, con varias rondas de ajuste de guías.
    - Herramienta: **Doccano**.
    - Evaluación de consistencia: **Fleiss' κ = 0.70–0.75** (acuerdo sustancial).
    - Diferenciación explícita entre:
        - **Skills**: aplicación de conocimientos (ej. "work independently").
        - **Knowledge**: saberes adquiridos (ej. "Python", "supply chain").
    - Se incluyen **actitudes** (ESCO → soft skills) bajo la categoría de skills.
3. **Estadísticas**:
    - Skills: medianas de 4–5 tokens.
    - Knowledge: típicamente ≤ 5 tokens.
    - Mayor acuerdo entre anotadores en *knowledge* que en *skills*.
    - Soft skills frecuentes: *proactive, communication skills, leadership*.
    - Knowledge frecuentes: *Java, Python, SAP, AWS*

c) Metodología experimental

1. **Formulación**: tarea de *sequence labeling* (BIO tags).
2. **Modelos probados**:
    - **BERTbase** (baseline).
    - **SpanBERT** (mejor manejo de spans largos).
    - **JobBERT**: BERT adaptado al dominio con 3.2M oraciones de JPs.
    - **JobSpanBERT**: SpanBERT con preentrenamiento adaptado a JPs.
    - **Longformer**: para secuencias largas (descartado por bajo rendimiento).
3. **Estrategias de aprendizaje**:
    - **STL**: modelos entrenados por separado en skills o knowledge.
    - **MTL**: un solo modelo entrenado para ambos tipos.
    - Capa final: **CRF** (Conditional Random Field).
    - Toolkit: **MACHAMP**.

d) Resultados

1. **Rendimiento global (F1 en spans)**
    - **Skills**: JobSpanBERT ≈ **56.6 F1** (mejor que BERTbase ≈ 53).
    - **Knowledge**: JobBERT ≈ **63.9 F1** (mejor que BERTbase ≈ 62.3).
    - **Combined**: JobBERT ≈ **59.7 F1**.
2. **Hallazgos principales**:
    - El **preentrenamiento continuo en textos de vacantes laborales** mejora consistentemente sobre modelos generales.
    - **STL > MTL**: entrenar skills y knowledge por separado supera al modelo conjunto.
    - **Span length effect**: peor desempeño en *skills cortas* (soft skills como "passionate"), mejor en *knowledge* cortos (tecnologías como "Python").
    - **Longformer** fue peor en todas las métricas → procesar postings completos no ayudó.
    - **Skills vs. Knowledge**: extraer skills es más difícil (más largas, ambiguas, semánticas); knowledge es más fácil (tokens concretos, p. ej. nombres propios).

e) Aportes clave

1. **Primer dataset público a nivel span** para extracción de habilidades con guías de anotación detalladas.
2. **Definición clara** de la relación entre knowledge (hard skills), skills (aplicación de knowledge) y actitudes (soft skills).
3. **Prueba empírica** de que **domain-adaptive pretraining** es esencial para SE.
4. **Relevancia práctica**: SKILLSPAN puede enriquecer taxonomías como ESCO con habilidades emergentes no contempladas.
5. **Limitaciones**: dataset aún pequeño (391 postings anotados), sesgo hacia inglés, diferencia marcada entre dominios (BIG vs TECH).

- Este paper es **la base de datos fundacional** que luego retoman trabajos como *Skill-LLM (Herandi et al. 2024)*.
- Refuerza la crítica a los enfoques clásicos de n-gramas/reglas → muestra que incluso con BERTbase se logran resultados modestos, y que la mejora viene de **embeddings adaptados al dominio**.
- Como vamos a concluir que a futuro seria bueno tener datasets anotados para hacerle finetuning a LLMs, nos aporta **guías de anotación concretas** que podemos citar para los que quieran construir o enriquecer tu dataset.
- Justifica la idea de diferenciar habilidades individuales (skill components) y no sólo mapear títulos de vacantes → porque los spans incluyen tanto tecnologías como actitudes blandas.
- Da soporte a nuestro pipeline de embeddings + clustering: ellos muestran que los spans son heterogéneos y requieren técnicas más semánticas para agrupar en perfiles de habilidades.

6. ESCOX: A Tool for Skill and Occupation Extraction Using LLMs from Unstructured Text

a) Qué hace

El artículo presenta **ESCOX (ESCO Skill Extractor)**, una herramienta **open-source** desarrollada en el marco del proyecto europeo **SKILLAB (Horizon Europe)**. Su propósito es identificar y clasificar **habilidades y ocupaciones** directamente desde textos no estructurados (p. ej., ofertas laborales, reportes, artículos).

A diferencia de la mayoría de sistemas que sólo extraen habilidades, **ESCOX realiza extracción dual (skills + occupations)**, mapeando los resultados a las taxonomías **ESCO** (skills/competences) e **ISCO-08** (ocupaciones). La motivación es reducir las brechas entre la oferta educativa y la demanda laboral, habilitando un análisis más fino de tendencias del mercado

b) Extracción y metodología

1. **Arquitectura técnica**
    - Backend: **Flask API** con endpoints REST (extracción de skills/occupations).
    - Modelo base: **Sentence Transformers (all-MiniLM-L6-v2)** → embeddings semánticos.
    - Matching: **cosine similarity** contra embeddings precomputados de ESCO/ISCO.
    - Umbrales ajustables (default: 0.6 para skills, 0.55 para occupations).
    - Almacenamiento: CSV con metadatos + BIN con embeddings precalculados.
    - Deployment: Docker Compose con Flask + Gunicorn + Nginx; también opción local.
2. **Pipeline de procesamiento**
    - Entrada: texto crudo (job posting, artículo, reporte).
    - Preprocesamiento: limpieza + (si es muy largo) **resumen con Summa**.
    - Extracción: embeddings → similitud → top matches ESCO/ISCO.
    - Salida: CSV estructurado con IDs ESCO/ISCO, descripciones y score de similitud.
3. **Interfaces**:
    - **GUI no-code** (subida de archivos o copy-paste, con visualización interactiva).
    - **API REST** para integración en pipelines.
    - **Google Colab** preconfigurado.
4. **Caso de estudio**:
    - Se recopilaron **6500 ofertas laborales (OJAs) de EURES** en el dominio de software engineering.
    - ESCOX extrajo ≈7400 habilidades y ≈6100 ocupaciones.
    - Skills top: *Java (27.7%)*, *SQL (19.2%)*, *DevOps (12.8%)*, *Work independently (10.1%)*, *Python (5.9%)*.
    - Occupations top: *ICT Business Analyst (35.2%)*, *Project Manager (23.7%)*, *Computer Scientist (15.6%)*

c) Resultados y rendimiento

1. **Eficiencia computacional**
   - CPU i7-14700F sin GPU → 472.49 s para 6500 postings (~13.8 postings/s).
   - GPU A100/L4 → ~130 s para 6500 postings (~50 postings/s).
   - Consumo memoria bajo (400–2000 MB).
2. **Precisión práctica**:
   - Alto rendimiento en detección de **hard skills técnicas**.
   - Soft skills detectadas, pero subrepresentadas en los postings.
   - Ocupaciones correctamente mapeadas a ISCO, incluso cuando no se mencionan explícitamente (p. ej., deducción por contexto).
3. **Comparación cualitativa con otros sistemas**
   - Frente a *SkillSpan* o *JobBERT*: añade integración taxonómica y GUI.
   - Frente a *SkillGPT*: menor "inteligencia libre" pero mayor parsabilidad y reproducibilidad.
   - Abierto (MIT License), modular, y reproducible → ventajas frente a sistemas cerrados.

d) Aportes clave

1. **Extracción dual** (skills + occupations) → puente entre competencias ("qué sabe hacer alguien") y funciones laborales ("para qué está calificado").
2. **Integración taxonómica completa** con ESCO e ISCO.
3. **Open-source, reproducible, modular**, fácilmente integrable en pipelines académicos y de política pública.
4. **Escalable y eficiente** incluso sin GPU, con benchmarks detallados.
5. **Orientación a usuarios no técnicos** gracias a su interfaz no-code y outputs CSV listos para análisis.

e) Limitaciones y futuro

- Actualmente se basa en **embeddings + cosine similarity** → menos preciso que LLMs fine-tuneados (ej. Skill-LLM).
- Dificultades en **soft skills contextuales** y en skills emergentes no incluidas en ESCO.
- Futuro: integrar **ANN (FAISS)** para acelerar búsquedas en grandes volúmenes, **transformers finos** para soft skills, y **estimación de incertidumbre** en los matches

- Refuerza que **taxonomías como ESCO son útiles pero insuficientes** → detecta bien lo que ya está en ESCO, pero limita la detección de nuevas habilidades emergentes.

7. Enhancing Job Posting Classification with Multilingual Embeddings and Large Language Models

a) Qué hace

El artículo aborda el **reto de clasificar ofertas laborales multilingües (español, italiano) contra la taxonomía ESCO en inglés**, enfrentando el problema de **solapamiento de ocupaciones y ambigüedad de skills**.

Proponen un modelo híbrido que combina:

- **Embeddings multilingües (E5-large)** para recuperar las definiciones de ocupaciones ESCO más cercanas.
- **Retrieval-Augmented Generation (RAG)** para enriquecer al LLM con contexto específico y reducir alucinaciones.
- **LLMs (Llama-3 8B, optimizado con Chain-of-Thought + DSPy)** para seleccionar títulos ocupacionales finales dentro de un conjunto restringido.

Además, exploran el uso de **LLMs como etiquetadores automáticos** (GPT-4o, Gemini 1.5, Claude 3.5) para construir un dataset de evaluación con anotaciones revisadas por humanos

b) Extracción y metodología

1. **Arquitectura**
    - Paso 1: embeddings multilingües (E5-large) → recuperación de top 30 ocupaciones ESCO más cercanas por similitud coseno.
    - Paso 2: construcción de un **prompt compuesto** con las descripciones ESCO recuperadas + la oferta de empleo.
    - Paso 3: LLM (Llama-3-8B) evalúa candidatos y selecciona el título más adecuado.
    - Restricción: salida limitada al conjunto de títulos válidos en ESCO (evita labels inventados).
2. **Dataset de prueba**
    - 200 ofertas reales (100 IT, 100 ES) de InfoJobs con descripciones largas.
    - Preprocesamiento: anonimización y limpieza.
    - Etiquetado inicial con GPT-4o, Gemini, Claude; luego validación humana.
    - Acuerdo entre LLMs: $\kappa \approx 0.62$ (sustancial).
3. **Baselines**
    - **SkillGPT** (embeddings + similitud directa).
    - **Zero-shot classification (mBART-MNLI, Llama-3)**: formulado como NLI.
    - **Multilingual E5-large embeddings** sin LLM.
4. **Optimización**
    - Usan **DSPy BootstrapFewShot** para optimizar Llama-3 con 10 ejemplos de entrenamiento y 30 de validación.
    - Profesor: GPT-4o (teacher-student optimization).

c) Resultados

1. **Italiano (Tabla 2)**
    - **Llama-3-8B (CoT optimizado)** → Precisión@5 = 0.32, Recall@5 = 0.76 (mejor).

    - **Embeddings E5-large** → Recall@10 = 0.88 (máximo recall).
    - **Baselines (SkillGPT, MNLI)** → significativamente inferiores.
2. **Español (Tabla 3)**
    - **Llama-3-8B (CoT optimizado)** → Precisión@5 = 0.28, Recall@5 = 0.72.
    - **Embeddings E5-large** → Recall@10 = 0.92 (máximo recall).
    - Again, SkillGPT y MNLI por debajo.
3. **Hallazgos clave**:
    - **LLM optimizado (CoT)** logra el mejor balance precisión–recall en top 5.
    - **Embeddings multilingües** maximizan recall en top 10, útiles como paso de filtrado.
    - Zero-shot (MNLI) funciona, pero con recall más bajo (0.58–0.70).
    - SkillGPT tiene desempeño débil, confirmando que embeddings solos no bastan.
4. **Costo computacional**
    - Llama-3-8B: ~1.5 s/posting, GPU 16 GB, alto consumo energético.
    - mBART-MNLI: <0.5 s/posting, 8 GB GPU.
    - E5-large: <0.2 s/posting en CPU (más eficiente y eco-friendly).

d) Aportes clave

1. **Multilingual embeddings + LLMs** superan ampliamente a métodos anteriores de clasificación ocupacional (ej. JobBERT, SkillGPT).
2. **RAG restringido a ESCO**: evita alucinaciones y asegura outputs parsables.
3. **LLMs como etiquetadores automáticos**: muestran acuerdo sustancial, útiles para bootstrap de datasets.
4. **Balance precisión vs recall**: LLM optimizado ideal para clasificación fina; embeddings ideales para recuperación masiva.
5. **Evaluación multilingüe real (IT, ES)**: novedad importante frente a trabajos previos centrados solo en inglés.

e) Limitaciones y futuro

- Dataset pequeño (200 postings) → se necesita ampliar.
- Costos computacionales del LLM limitan despliegue en producción.
- Ambigüedad en ocupaciones con solapamiento semántico persiste (ej. *Project Engineer* vs *Project Manager*).
- Futuro: aplicar la metodología también a **perfiles de candidatos (CVs)** y estudiar **efecto de lexical overlap multilingüe**

- Este paper **valida El enfoque híbrido**: embeddings → LLM → clasificación.
- Aporta evidencia de que **embeddings multilingües son clave para recall alto**, mientras que los **LLMs refinan precisión**.
- Refuerza la crítica a la dependencia exclusiva de taxonomías (ESCO/CIUO): muestran solapamientos problemáticos.
- El uso de **LLMs como etiquetadores automáticos** puede inspirar a futuro propio dataset de validación para skills.

8. Multilingual Skill Extraction for Job Vacancy–Job Seeker Matching in Knowledge Graphs

a) Qué hace

El artículo propone un **framework multilingüe basado en grafos de conocimiento** para mejorar el **matching entre vacantes y candidatos** mediante la extracción precisa de habilidades, tanto de **job postings** como de **CVs**, y su alineación con la taxonomía **ESCO**.

La motivación central es que la mayoría de enfoques anteriores se limitan a **features superficiales** (geografía, clics, títulos), o bien usan LLMs sin aprovechar la **estructura jerárquica y multilingüe de ESCO**, lo que genera sesgos, alucinaciones o baja interpretabilidad

Este framework busca combinar lo mejor de tres mundos:

1. **Extracción robusta de habilidades** (varios métodos en paralelo).
2. **Filtrado y normalización** usando prompts con Chain-of-Thought (CoT) y DSPy.
3. **Representación en grafo de conocimiento multilingüe**, optimizado con embeddings (ej. español-inglés) y relaciones jerárquicas de ESCO.

b) Extracción y metodología

1. **Fuentes y preprocesamiento**
    - Datos: **InfoJobs** (648 vacantes, 1200 CVs, subset en español).
    - Limpieza: anonimización, eliminación de contenido promocional o sensible.
    - Normalización de títulos: CoT prompting convierte títulos genéricos (ej. *administrativo*) en específicos únicos con índices, evitando duplicados en el grafo.
2. **Extracción de habilidades (3 métodos en paralelo)**
    - **Entity linking (BLINK adaptado a ESCO)** → mapea spans directamente a descripciones de ESCO.
    - **Extreme multi-label classification (IReRa + DSPy)** → LLMs que iteran recuperación–ranking.
    - **Few-shot ICL con LLMs (Nguyen et al. 2024)** → prompts multi-ejemplo para capturar skills largas/ambiguas.
3. **Skill selection y mapeo a ESCO**
    - Combinación de salidas + módulo CoT con “hints” en DSPy.
    - Traducción automática a inglés (skills detectadas en español).
    - Integración de embeddings multilingües (E5) + descripciones ESCO en base vectorial para mejorar recall y contexto.
4. **Construcción de Knowledge Graph**
    - Nodos: vacantes, candidatos, habilidades ESCO.
    - Aristas: relaciones jerárquicas (padre–hijo) + coincidencias n-gram.

- Pesos iniciales: 1 para padres, 0.5 para hijos → refinados con **IDF** (diferencia entre skills comunes y especializadas).
- Refinamiento: **Node2Vec + random walks sesgados** sobre embeddings multilingües (ej. fastText ES + Word2Vec).
- Resultado: representación vectorial del grafo que permite comparar candidatos–vacantes en espacio semántico.

c) Resultados

1. **Extracción de habilidades**
   - **ICL Claude-3.5 Sonnet** → mejor recall (0.50) pero baja precisión.
   - **DSPy optimizado con Llama-3** → mejor balance, F1 ≈ 0.35–0.43.
   - En general, recall sube con ICL, pero precision baja (más falsos positivos).
2. **Matching vacante–candidato (grafo)**
   - **IReRa baseline** → Jaccard 0.308.
   - **BLINK baseline** → Jaccard 0.416.
   - **Modelo final (skills filtradas + grafo + embeddings ES/EN)** → Jaccard 0.5002, Cosine similarity 0.5761.
   - Esto muestra mejora significativa respecto a baselines de extracción aislada.
3. **Validación**
   - Dataset anotado manualmente por 12 expertos.
   - Skills etiquetadas como *essential* o *irrelevant*.
   - Combinación humano + Claude-3.5 para aumentar cobertura y mitigar sesgos.

d) Aportes clave

1. **Extracción multimétodo** → combina entity linking, XMC, y LLM few-shot para mayor robustez.
2. **Normalización con CoT** → asegura consistencia en skills y títulos.
3. **Integración jerárquica + IDF** en el grafo → pondera importancia de skills comunes vs especializadas.
4. **Embeddings multilingües** → puente entre vacantes en español y ESCO en inglés.
5. **Knowledge Graph enriquecido** → permite matching más fino y explicable entre vacantes y CVs.

e) Limitaciones y futuro

- La generación de **experiencias sintéticas con LLMs** puede introducir sesgos si no se controla

9. Rethinking Skill Extraction in the Job Market Domain using Large

Language Models

a) Qué hace

El artículo explora el uso de **LLMs (GPT-3.5, GPT-4)** para dos tareas críticas en el mercado laboral:

Clasificación de ocupaciones (mapear ofertas a taxonomías como ESCO).

Extracción de habilidades (hard y soft skills desde descripciones textuales).

La hipótesis es que los **LLMs zero/few-shot pueden simplificar pipelines complejos** basados en embeddings, NER y reglas, ofreciendo un enfoque unificado para clasificación y extracción (PARECIDO A NOSOTROS)

b) Metodología

1 **Datos**

- Fuente: ofertas de empleo recolectadas de portales nórdicos y europeos.
- Subconjunto multilingüe (inglés, sueco, finés).
- Se usaron anotaciones manuales de ocupaciones y habilidades como ground truth.

2 **Tareas**:

- **Clasificación ocupacional**: mapping a ESCO/ISCO.
- **Extracción de skills**: identificar spans dentro de la oferta, luego mapearlos a ESCO.

3 **Modelos probados**:

- **GPT-3.5 Turbo** y **GPT-4** (API OpenAI).
- Baselines:
  - **mBERT + CRF** para extracción.
  - **Sentence-BERT + cosine similarity** para clasificación.

4 **Prompts diseñados**:

- **Zero-shot**: instrucciones simples (“extrae habilidades en JSON…”).
- **Few-shot**: ejemplos de 3–5 postings anotados.

- Salida forzada en formato JSON estructurado.

5 **Evaluación**:

- Extracción: métricas span-level (F1 exacta).
- Clasificación: precisión Top-1, Top-5.

c) Resultados

1 **Clasificación de ocupaciones**

- **GPT-4 (few-shot)**: Precisión Top-1 = **71%**, Top-5 = **88%**.
- **GPT-3.5**: Top-1 = 58%, Top-5 = 79%.
- **Baseline Sentence-BERT**: Top-1 = 62%, Top-5 = 82%.
- → GPT-4 supera al baseline, especialmente en precisión Top-1.

2 **Extracción de habilidades**

- **GPT-4 (few-shot)**: F1 = **0.62** (skills), **0.67** (knowledge).
- **GPT-3.5**: F1 ≈ 0.50–0.55.
- **Baseline mBERT-CRF**: F1 ≈ 0.57.
- → GPT-4 logra mejor performance que modelos entrenados específicamente, sin necesidad de fine-tuning.

3 **Hallazgos adicionales**:

- Few-shot mejora significativamente frente a zero-shot (≈+10 F1 y +12% precisión Top-1).
- Outputs de GPT-4 fueron **parsables y consistentes** en formato JSON, aunque aún con casos de alucinación.
- Multilingüe: desempeño cae en sueco/finés (≈−12% F1), pero se mantiene competitivo.

d) Aportes clave

1 **Demostración empírica** de que LLMs generalistas (sin fine-tuning) pueden competir con modelos entrenados en dominio laboral.

2 **Unificación de pipeline**: un mismo modelo puede realizar clasificación y extracción, reduciendo complejidad técnica.

3 **Flexibilidad multilingüe**: resultados aceptables incluso fuera del inglés.

4 **Ventaja práctica**: evita necesidad de datasets grandes, costosos de anotar.

5 **Limitaciones**:

- Costos altos de API (en inferencia masiva).
- Alucinaciones en skills raras.
- Menor rendimiento en lenguas de baja representación.

- Refuerza la crítica a los enfoques clásicos (n-gramas, TF-IDF, CRFs): GPT-4 sin entrenamiento específico ya los supera.
- Nos sirve como argumento de que **los LLMs son viables incluso sin fine-tuning**, siempre que usemos prompts bien diseñados y formatos controlados (JSON).

MEDICIONES Y COMPARACIONES

Mini-Gold (humano), 300 anuncios, Muestra: 300 postings (95% de confianza, ±5–6 p.p. para proporciones típicas).

Silver: 10-20k ofertas, match exacto con regex a esco + habilidades top, o embeddings a extraidas vs embeddings de habilidades y match exacto de .85 upwards se asume como bien.