OVERTURE MAPS
FOUNDATION

Indira Mariya

Crown 102

December 4, 2025

Project C places conflation

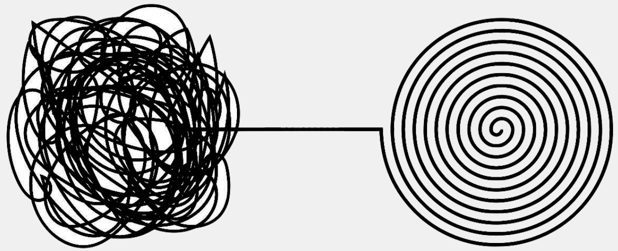
Problem Statement

Determine if small, scalable Language Models can perform better than the current places classifier.

1. Improve Accuracy: Can a model achieve higher accuracy in matching places data?

2. Improve Efficiency: Can a model offer a better price-to-performance ratio than the existing matcher?

Problem Breakdown



Analyzing Cost:

Time it takes to run – how long the full evaluation requires

Tokens used per match – how many LLM tokens are consumed for each comparison

Cost per match – approximate monetary cost for processing each pair

Analyzing Performance:

1. **Accuracy** – how many predictions were correct overall
2. **Precision** – of the predicted matches, how many were truly matches
3. **Recall** – of the true matches, how many were found by the model
4. **F1 Score** – balanced measure combining precision and recall

Constraints + Challenges

Must work at large scale – system needs to process thousands of POIs

Must stay within budget – model choice and token usage must remain cost-effective

Place data is noisy – spelling, formatting, and abbreviations vary widely

Global variation makes matching harder – addresses differ across countries and languages

Objective:

Replace Overture’s existing place matcher with a scalable, language-model based system that improves matching precision and cost efficiency.

01

Key Result

Train and evaluate at least three scalable models on the dataset given.

02

Key Result

Reduce average cost per inference by ≥15%, as measured using compute time and API/token usage logs.

03

Key Result

Reduce false matches by ≥10% relative to the overture matching model’s baseline metrics.

OKR Completion

KR1 — Passed

Evaluated four different scalable models, exceeding the requirement of three.
Models were tested using a consistent validation pipeline, allowing direct comparison.

KR2 — Passed

Achieved a major reduction in cost per inference.
The best-performing model (llama-3.1-8b-instant) costs significantly less than many other larger models.
Strong performance despite dramatically lower cost makes this KR fully met.

KR3 — Almost Met

None of the tested models reached or exceeded Overture's baseline precision of 0.93.
Primary limitation: smaller and mid-size LLMs struggle with borderline cases and subtle address distinctions, which Overture's established model handles better.
With further tuning, prompting, or ensemble methods, this KR could become reachable.

Approach

Data Cleaning

01

Data cleaning ensures all fields follow a consistent format so the matching models can compare records reliably.

It reduces noise, fixes inconsistencies, and prepares the data for evaluation.

Similarity Score

02

Similarity scoring provides an initial estimate of how similar two POI records are.

It serves as a fast baseline classifier before the LLM performs deeper reasoning.

LLM Matching

03

Use a language model to validate or correct the fuzzy match by checking the key address fields.

It outputs a JSON decision with a same_place label, confidence score, and a brief explanation.

Evaluate Results

04

Evaluate LLM predictions using accuracy, precision, and recall, then review incorrect matches to understand issues.

Data Cleaning

- Expand abbreviations
  - st = street
  - ave = avenue
  - cir = circle
  - N = north
  - apt = apartment
- convert to lowercase
- remove extra characters from phone number (such as + and ())
- remove the https: from websites
- Example normalized data: {""freeform"":""Avenida Carlos Cano"",""locality"":""Los Barrios"",""postcode"":""11370"",""country"":""ES""}
- **avenida carlos cano,los barrios,,11370,ES**

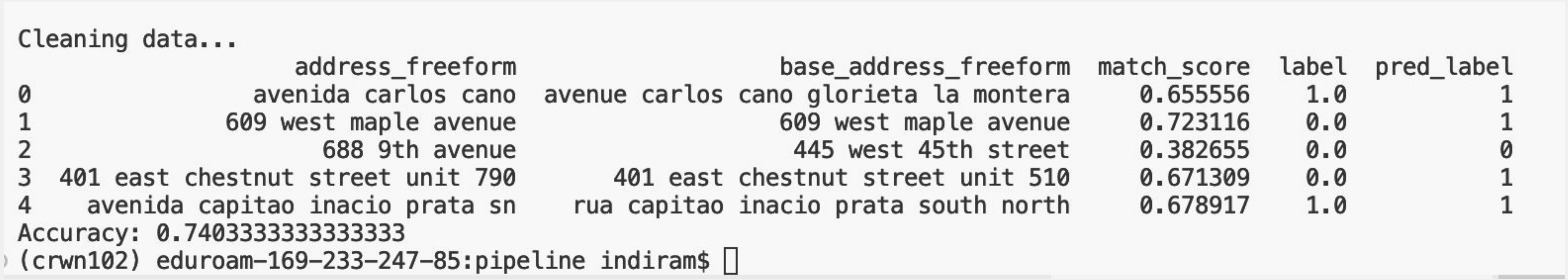
Fuzzy Match Score

similarity =
20% name_sim
50% address_sim
15% phone_sim
5% website_sim
10% category_sim

- Calculate a fuzzy matching similarity score for each field in each row
- RapidFuzz token_sort_ratio
  - Compares words ignoring order (e.g., "Starbucks Coffee" ≈ "Coffee Starbucks").
- RapidFuzz partial_ratio
  - Handles substring overlap (e.g., "McDonalds" ≈ "McDonalds Hamburgers").
- RapidFuzz token_set_ratio
  - Handles cases with repeated/extra tokens (e.g., "CVS Pharmacy" ≈ "CVS").
- Jaro–Winkler similarity
  - Character-level similarity robust to typos (e.g., "Walgreens" ≈ "Walgreans").
- Sum up scores for each field for a total similarity score

Base Accuracy

- Using just a fuzzy similarity score, and choosing 1 or 0 based on a threshold of 0.671 we get a base accuracy = ~74%



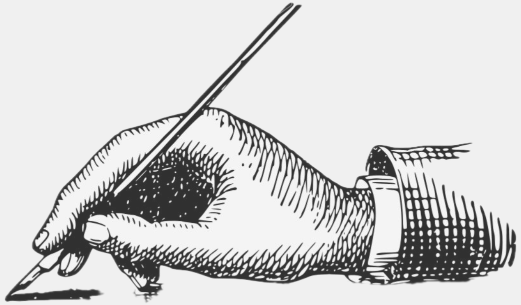
```
Cleaning data...
                   address_freeform                      base_address_freeform  match_score  label  pred_label
0               avenida carlos cano  avenue carlos cano glorieta la montera     0.655556    1.0           1
1            609 west maple avenue                      609 west maple avenue     0.723116    0.0           1
2                    688 9th avenue                       445 west 45th street     0.382655    0.0           0
3  401 east chestnut street unit 790       401 east chestnut street unit 510     0.671309    0.0           1
4    avenida capitao inacio prata sn    rua capitao inacio prata south north     0.678917    1.0           1
Accuracy: 0.7403333333333333
(crwn102) eduroam-169-233-247-85:pipeline indiram$ 
```

Prompt Creation

1. PRIORITIZE "sim" AND "pred":
   - If pred = 1 and sim is high (>= 0.7), assume same_place = 1 unless the address clearly disagrees.
   - If pred = 0 and sim is low (<= 0.3), assume same_place = 0 unless the address clearly shows they match.

2. USE THE ADDRESS AS A CONFIRMATION CHECK (SECONDARY):
   - Address fields are already normalized.
   - If the baseline decision from sim + pred conflicts with the address (e.g., different street numbers or different cities), flip the decision.

3. GENERAL RULE:
   - Similar address → supports same_place = 1
   - Very different address → supports same_place = 0
   - When uncertain, rely on sim + pred unless the address strongly contradicts it.



Prompt Cont.

```
OUTPUT REQUIREMENTS:
- Return ONLY a JSON object.
- "same_place" must be 1 or 0.
- "confidence" must be a number between 0.0 and 1.0.
- "explanation" must be 1 sentence and MUST describe the specific fields
  that led to the decision.
- Do NOT reuse generic phrases across answers.

RECORD A:
Address (freeform): {address_freeform}
Locality / City:    {address_locality}
Region / State:     {address_region}
Postcode:           {address_postcode}
Country:            {address_country}
Phone:              {phone}

RECORD B:
…..
```

Outputs

llama-3.1-8b-instant

- The inputs are fed into the model, and we are given an output in the format we specified
- Explanation is added for further context on why the model made its decision
- can help finetine the prompt to account for mistakes

Example 1

avenida carlos cano,los barrios,
avenue carlos cano glorieta la montera,los barrios

{'same_place': 1, 'confidence': 0.9, 'explanation': 'The decision to consider these records as the same place was made because the **street name, locality, postcode, and country** matched, and although the phone numbers were not identical, the **similarity score** and the matching address components supported this conclusion.'}

Example 2

1100 west stewart drive,orange,ca
1000 west la veta drive 1st floor,orange,ca

{'same_place': 0, 'confidence': 0.6, 'explanation': "The decision was made because the **street numbers and street names** in the address fields of the two records do not match, with record A having '**1100 west stewart drive**' and record B having '**1000 west la veta drive 1st floor**', which indicates they are likely different places."}

Outputs...

Translation Issue: Guessed 0, Actual: 1

avenida **capitao inacio prata sn**,sao goncalo do amarante,ce,62670000,BR,,

rua **capitao inacio prata south north**,,,62670000,BR,8533156140.0,0.71042148114427

Reasoning: "The decision was based on the conflicting street types and names, with 'avenida' in Record A and 'rua' in Record B, as well as the missing locality and region in Record B, which outweighed the matching postcode and country."

Address Overlap Issue: Guessed: 1, Actual: 0

apollo children hospital no 15 shafee mohammed road thousand lights,chennai,tamil nadu,600006,IN,8605001066.0,

apollo hospitals greams road no 21 greams lane off greams road,chennai,tamil nadu, 600006,IN,8605001066.0,0.5454377901122027,1,0.8,

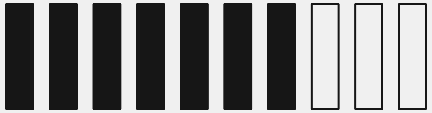
Reasoning: "The decision to consider these records as the same place was made because the phone number, locality, postcode, and country matched exactly between the two records, despite some differences in the address freeform field."

Results

Model Token Costs
1. llama-3.1-8b-instant — **$0.31** per 1M
2. llama-4-scout-17b — $**0.85** per 1M
3. llama-3.3-70b — **$5.17** per 1M
4. kimi-instruct — **$9.45** per 1M
5. kimi-0905 — **$13.42** per 1M

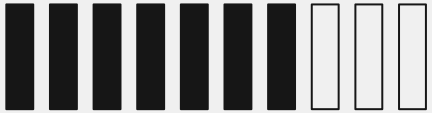
Model Speeds:
1. kimi-k2-instruct-0905 — 0.222s avg (4.50 matches/sec)
2. kimi-k2-instruct — 0.264s avg (3.78 matches/sec)
3. llama-3.1-8b-instant — 0.413s avg (2.42 matches/sec)
4. llama-4-scout-17b-16e-instruct — 1.083s avg (0.92 matches/sec)

Accuracy



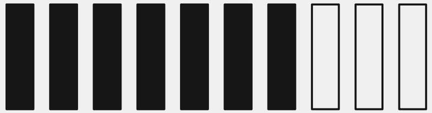
77%

llama-3.1-8b-instant
Accuracy: 0.77
Precision: 0.7651
Recall: **0.912**
F1 Score: 0.8321



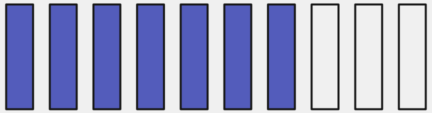
79%

kimi-k2-instruct-0905
Accuracy: 0.795
Precision: 0.8182
Recall: 0.864
F1 Score: 0.8405



78%

llama-4-scout-17b-16e
-instruct
Accuracy: 0.785
Precision: **0.8534**
Recall: 0.792
F1 Score: 0.8216



80%

kimi-k2-instruct
Accuracy: **0.800**
Precision: 0.8195
Recall: 0.872
F1 Score: **0.845**

Overall Best

llama-3.1-8b-instant

1. Highest recall (0.912)
finds the most true matches, minimizes missed duplicates.

2. Strong F1 score (0.8321)
good balance of precision and recall.

3. Lowest cost ($0.31 per 1M tokens)
much cheaper than other tested models

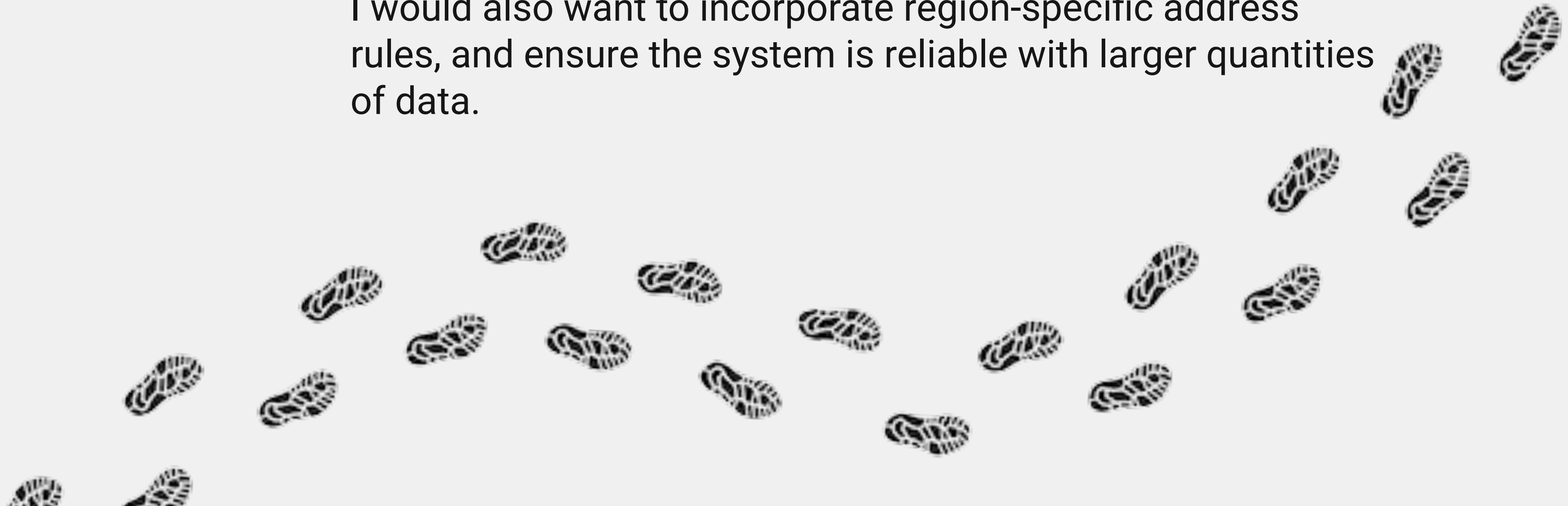
4. Fast runtime (0.413s avg)
efficient for large-scale processing, scalable

5. Stable precision (0.7651)
good reliability without excessive false positives

Next Steps

To continue this project, I want to tune the parameters even more and refine the model prompts to reduce common errors.
I also plan to experiment with combining multiple model outputs maybe using a framework like XGBoost to create a ensemble classifier.

I would also want to incorporate region-specific address rules, and ensure the system is reliable with larger quantities of data.



Thank you

Do you have any questions?

Github Repo