Extracting Graph Relations from Natural Language in Historical *Emigrant Savings Bank* Data

Stephen Balogh
Chun Ming (Sam) Ho

Emigrant Industrial Savings Bank

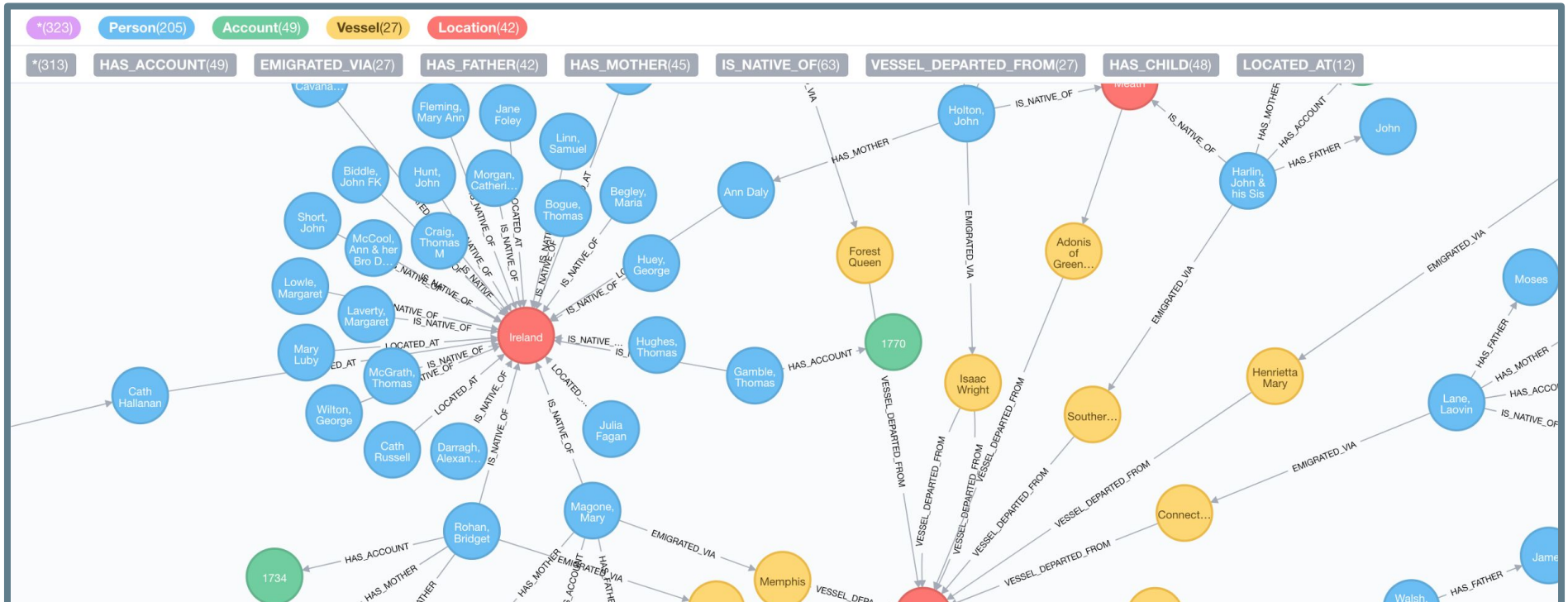
- Founded in NYC in 1850
- Bank records have been hand transcribed, we have access to 25k accounts
- Each account has a “Remarks” field that (usually) contains many pieces of information about the account holder:
  - Country / location of origin
  - Occupation data
  - Account of emigration to U.S. (shipnames, departure locations, dates)
  - Family tree (where parents / siblings are located currently and whether or not they are alive)
  - Details about marriages
- The dataset is of interest to historians working on New York City

Our Project

Turning a natural language field like this:

`Native of Cormack, 7mi from Loughrea, Co Galway Ire - Arr NY May 07 per the ship Howard from L'Pool - Parents Dead, Fa Owen- Mo Mary Quill, 2 Sisters Marg in NY- Bridget in Ire -Is Single`

... into entities and relationships that can populate a graph network:

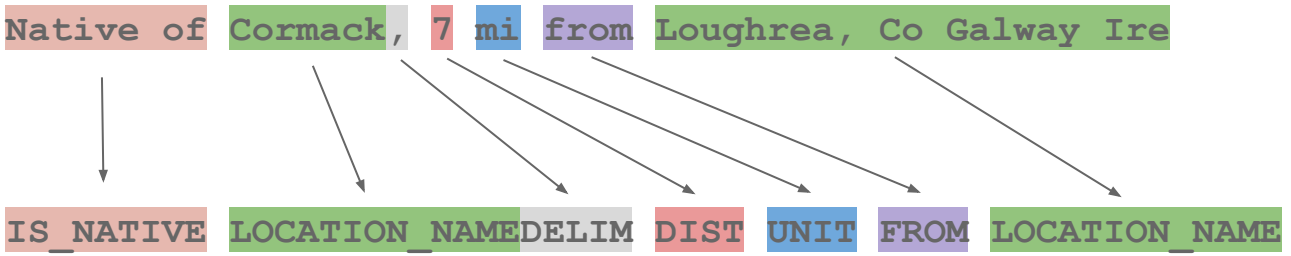


Relevant Entities / Relations

- Existing (non-NLP) work has been done in trying to make sense of this data
  - Historians are interested in capturing any named entities that appear in the “remarks” field, particularly those related to emigration
- We looked at the dataset, and some of the work that has been done in converting it to a database, and came up with a set of entity **types** and **relationships** that occurred with enough regularly that we thought we could capture them using NLP
- These entities and relationships comprise the vast majority of the content of the “remarks” fields that we looked at
- We then started to construct labels that would help in identifying these features

1. Labeling Token Sequences

- Unfortunately, none of the data was labeled, so we had to construct a training set ourselves
  - But, it gave us freedom to choose whatever labels we wanted
- We used conditional random fields (CRF) in two consecutive passes to predict labels on a per-token basis:
  - First pass uses labels to semantically categorize general **types of statements** that might be found within the “remarks” text field
    - E.g. every token in `Native of Cormack, 7 mi from Loughrea, C Ire` would be tagged with “subj:native-of”, since that is the “theme” under which this part of the text falls under
  - Second pass then uses these theme tags, along with other features, to map tokens into symbols expressing relationships, units, named entities, and etc



1. Labeling Token Sequences *(cont.)*

- CRF is a discriminative model, so we have to extract features from our sequence of input tokens
- We used the sklearn-crfsuite implementation of CRF, so all of our feature extraction happened in Python
- Our features encoded:
  - Word shape
  - Prefix / suffix bigrams and trigrams
  - Presence/absence of non-alphabetic characters
  - Presence of an element belonging to a group of known, meaningful tokens (e.g. “lre” or “LP”, which are known location abbreviations)
  - A couple syntactic features (if token is found within open brackets or parens)
  - ...
- Trained with gradient descent
- Model performs surprisingly well even with only 81 annotated training examples; on random sample of 30 record predictions:
  - Accuracy of “theme” tags was `1.0`
  - Accuracy of per-token symbols was `0.97`

Token	Statement Label (CRF 1)	Token Tag (CRF 2)
She	subj:native-of	t:subj:IS_WOMAN
Nat	subj:native-of	t:rel:IS_NATIVE_OF
of	subj:native-of	t:rel:IS_NATIVE_OF
Ferrymount	subj:native-of	t:location:NAME
,	subj:native-of	t:DELIMITER
6	subj:native-of	t:location:DISTANCE
miles	subj:native-of	t:location:DISTANCE_UNIT
from	subj:native-of	t:location:FROM
Mt	subj:native-of	t:location:NAME
Mellick	subj:native-of	t:location:NAME
,	subj:native-of	t:location:NAME
Queens	subj:native-of	t:location:NAME
,	subj:native-of	t:location:NAME
Ire	subj:native-of	t:location:NAME
-	delimiter:thematic	delimiter:thematic
Arr	subj:emigration-event	t:emigration:ARRIVED
Jul	subj:emigration-event	t:time:MONTH
1844	subj:emigration-event	t:time:YEAR
per	subj:emigration-event	t:emigration:VIA
Fairfield	subj:emigration-event	t:emigration:VESSEL
from	subj:emigration-event	t:emigration:VESSEL_HAS_ORIGIN
LP	subj:emigration-event	t:location:NAME
...	...	...

Sequence subset of tuples containing *token*, *statement*, *tag*

2. Interpreting Label Sequences

“Nat of Ferrymount, 6 miles from Mt Mellick, Queens, Ire - Arr NY per ship Orinthian fr London - Par dead Fa Jno,Mo Sarah ...”

- We used context free grammar (CFG) to parse each group of statement-label tokens into parse tree.

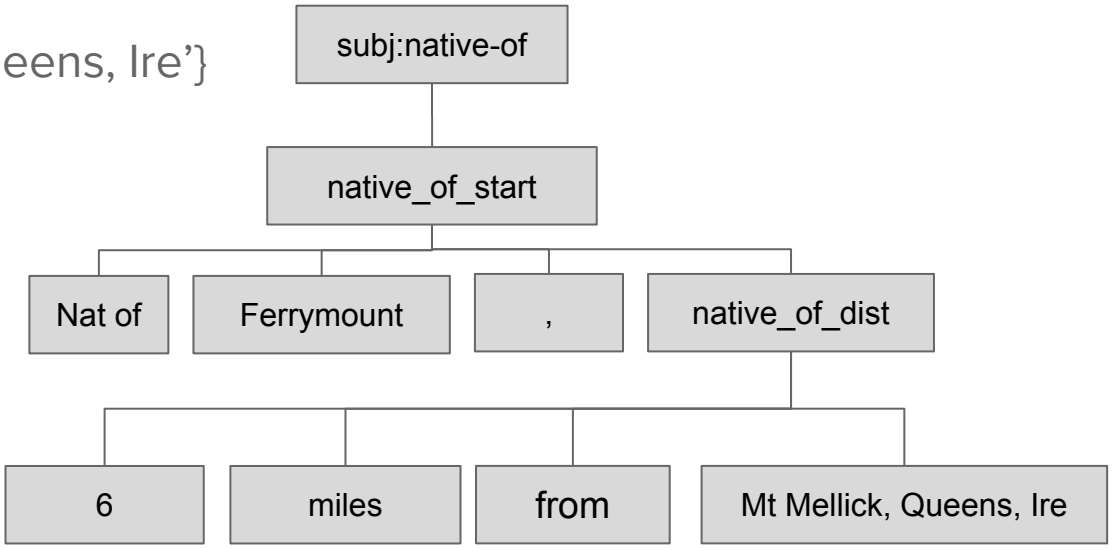
subj:native-of	
Nat of	t:rel:IS_NATIVE_OF
Ferrymount	t:location:NAME
,	t:DELIMITER
6	t:location:DISTANCE
miles	t:location:DISTANCE_UNIT
from	t:location:FROM
Mt Mellick, Queens, Ire	t:location:NAME

LHS	**RHS**
native_of_start	[t:rel:IS_NATIVE_OF] [t:location:NAME] [t:DELIMITER?] [native_dist?] [t:location:NAME?]
native_of_dist	[t:location:DISTANCE] [t:location:DISTANCE_UNIT?] [t:location:FROM] [t:location:NAME]

2. Interpreting Label Sequences *(cont.)*

- We use Breadth-First-Search (BFS) in each statement-label subtree and Depth-First-Search (DFS) for each named entity.

{‘location’, ‘Ferrymount’}
{‘distance_from’,
… {‘from’, ‘Mt Mellick, Queens, Ire’}
… {‘distance’, ‘6 miles’}



```
{
  "native_of": {
    "location": "Curraghamore , Co Donegal"
  },
  "emigration": [
    {
      "date": {
        "month": "July",
        "date": "6",
        "year": "1859"
      },
      "vessel": {
        "vessel": "James Foster",
        "location": "LP"
      }
    }
  ],
  "parents": [
    {
      "type": "Father",
      "name": "Jas"
    },
    {
      "type": "Mother",
      "name": "Mary Hannegan"
    }
  ],
  "martial": {
    "status": "Is Single"
  }
...
```


Sample of a final “extracted” record

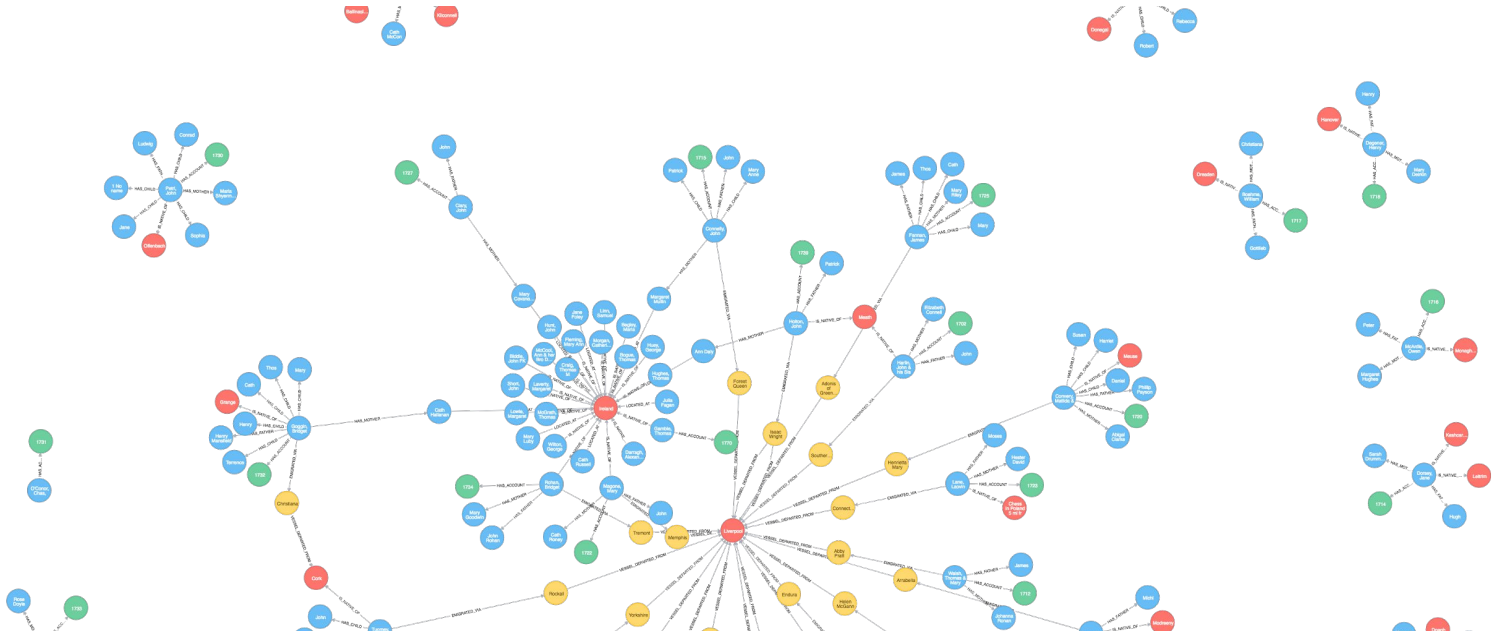
End-to-end metrics

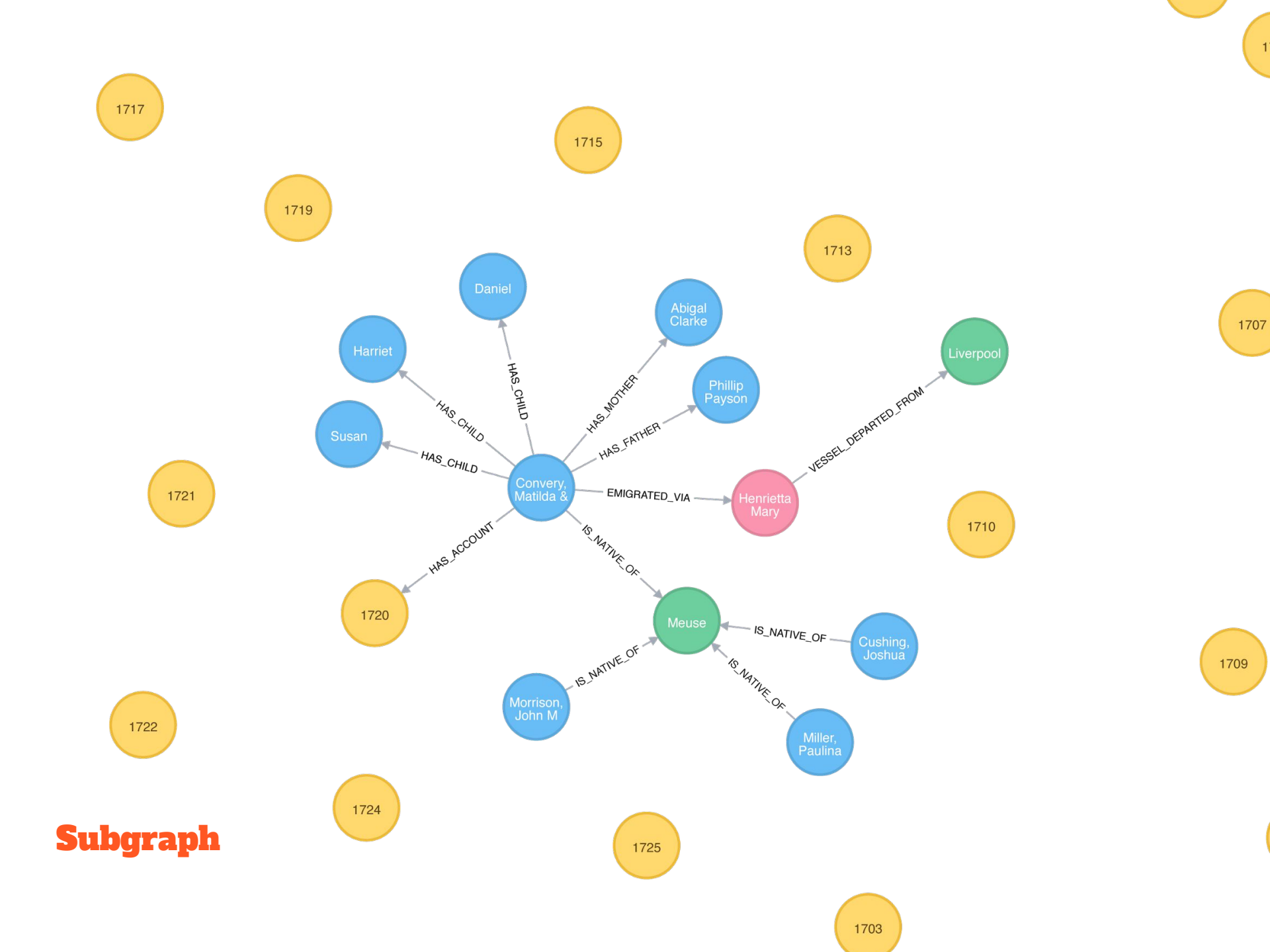
- We decided to use precision and recall (plus F_1 scores)
  - Take a random sample of records that did not appear in the training data
  - Run them through CRF models, then CFG-based (and heuristic) interpreters
  - Compare claims made in the resulting extraction to the total amount of human-detectable correct claims from the original text
    - Possible claims come from a bounded domain of possibilities including:
      - Existence and related properties for Locations, Persons, Ships, Voyages, Employers
      - Relationships between entities such as “is_brother_of”, “emigrated_via”, “native_of”, etc.
    - The claims we can support are bounded by the CRF labels we created and the sequences that we designed our CFG around
  - From a random sample of 30 records that we labeled to use as our dev set:

Metric	*Non-CFG Interpreter*	*CFG Interpreter*
Precision	0.9424	**0.9612**
Recall	0.3669	**0.7647**
F_1	0.5282	**0.8517**

Building a sample graph

- From our extracted record schema, it is straightforward to actually construct a graph network
- Some standardization needs to happen, particularly regarding **locations**
  - Location names are ambiguous, or may appear in many different formats
  - We geocoded our location strings and tied them to an authority (GeoNames/WOF)
  - Used the Elasicsearch-based Pelias geocoder
- We used Neo4j to build a proof-of-concept graph network




1717
1715
1719
1713
1707
1721
1710
1709
1722
1724
1725
1703
Daniel
Abigal Clarke
Harriet
Liverpool
Phillip Payson
Susan
Convery, Matilda &
Henrietta Mary
1720
Meuse
Cushing, Joshua
Morrison, John M
Miller, Paulina
HAS_CHILD
HAS_CHILD
HAS_CHILD
HAS_MOTHER
HAS_FATHER
VESSEL_DEPARTED_FROM
EMIGRATED_VIA
HAS_ACCOUNT
IS_NATIVE_OF
IS_NATIVE_OF
IS_NATIVE_OF
IS_NATIVE_OF
Subgraph

Thanks!