



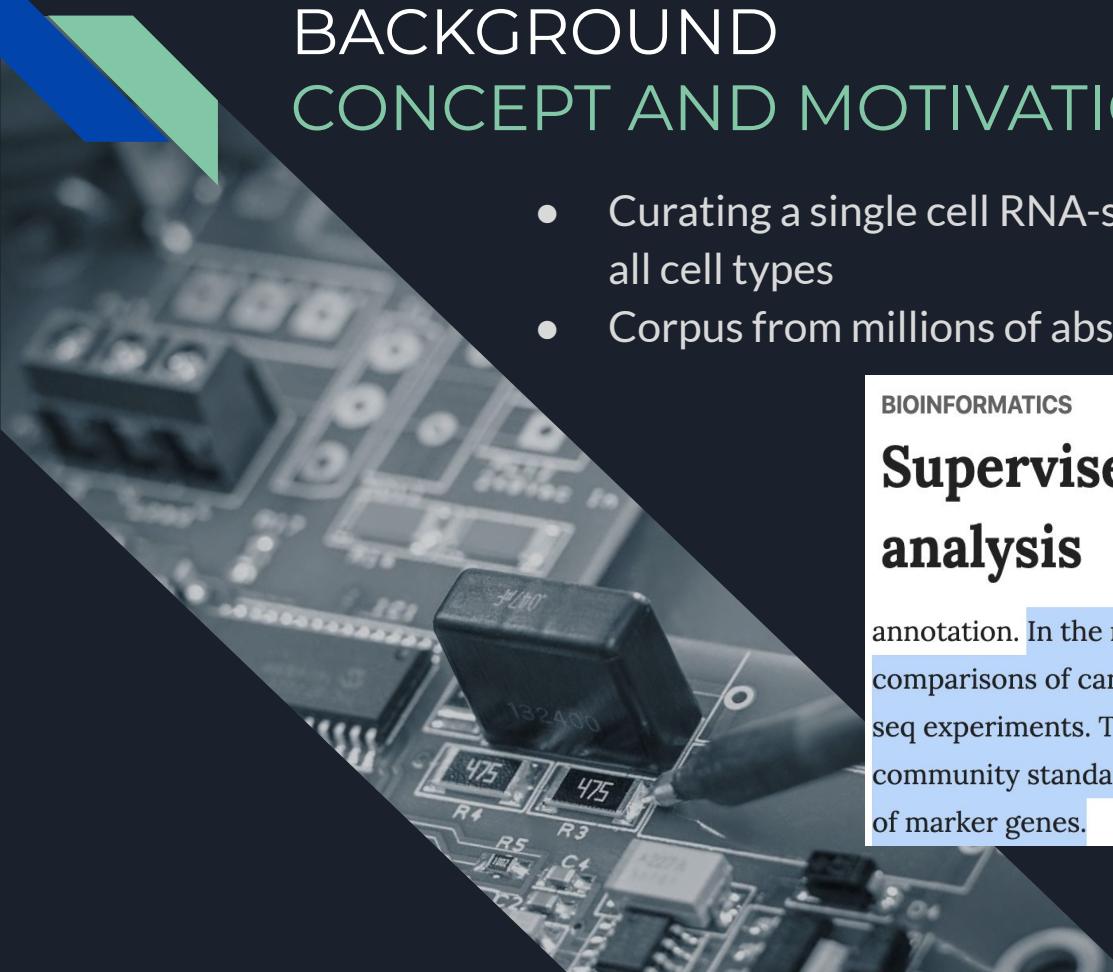
# TYPER: Presentation

Steven Wu  
Nicolas Cardozo  
Samantha Borje



# BACKGROUND CONCEPT AND MOTIVATION

- Curating a single cell RNA-sequencing markers from across all cell types
- Corpus from millions of abstracts from pubmeds API



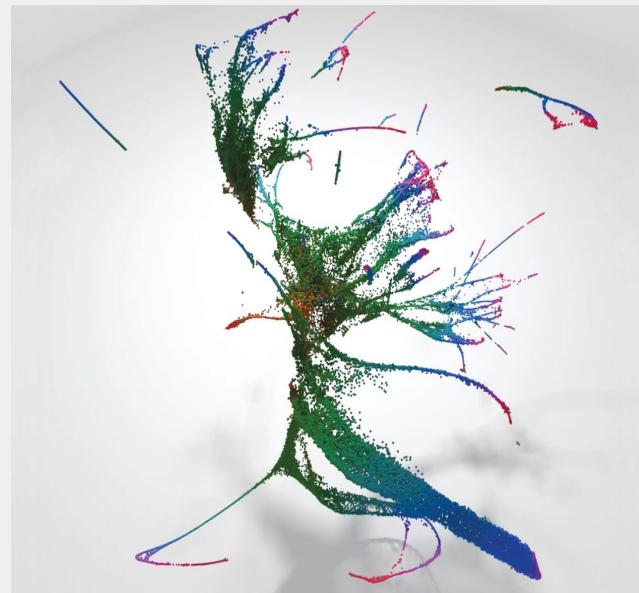
BIOINFORMATICS

## Supervised clustering for single-cell analysis

annotation. In the near future, we are likely to see large scale comparisons of candidate lists derived from the literature and scRNA-seq experiments. This will, in turn, drive the emergence of consistent community standards regarding the selection, evaluation and reporting of marker genes.

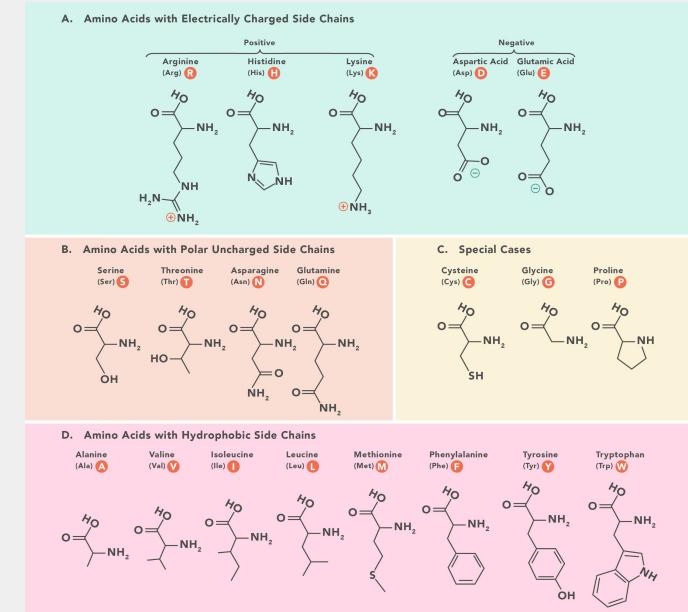
# RESULTS USE CASE

- The problem we are trying to solve is curating a list of single cell RNA-sequencing markers from across all cell types.
- We are going to curate the corpus from millions of abstracts from pubmeds API.



# RESULTS

## PROOF OF CONCEPT

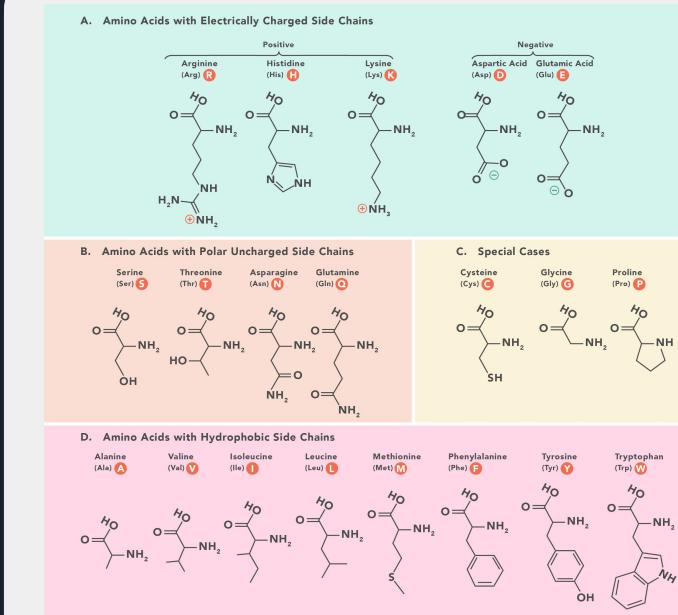


```
# load libraries
from gensim.models import
KeyedVectors
import numpy as np
import matplotlib.pyplot as plt
import umap
import pandas as pd
import seaborn as sns

# load word embeddings
word_vectors =
KeyedVectors.load_word2vec_format('pubmed2018_w2v_200D/pubmed2018_w2v_200D
.bin', binary=True)
```

# RESULTS

## PROOF OF CONCEPT



```
# 20 AMINO ACIDS to query and their properties
# define marker/shapes for each amino acid based off of property
aa = ['alanine', 'isoleucine', 'leucine', 'methionine',
      'valine', 'phenylalanine', 'tryptophan', 'tyrosine',
      'asparagine', 'cysteine', 'glutamine', 'serine',
      'threonine', 'aspartic-acid', 'glutamic-acid', 'arginine',
      'histidine', 'lysine', 'glycine', 'proline']

aa_properties = ['Hydrophobic', 'Hydrophobic', 'Hydrophobic', 'Hydrophobic',
                 'Hydrophobic', 'Hydrophobic', 'Hydrophobic', 'Hydrophobic',
                 'Polar', 'Special', 'Polar', 'Polar',
                 'Polar', 'Negative', 'Negative', 'Positive',
                 'Positive', 'Positive', 'Special', 'Special']

aa_short = ['A', 'I', 'L', 'M', 'V', 'F', 'W', 'Y',
            'N', 'C', 'Q', 'S', 'T', 'D', 'E', 'R',
            'H', 'K', 'G', 'P']

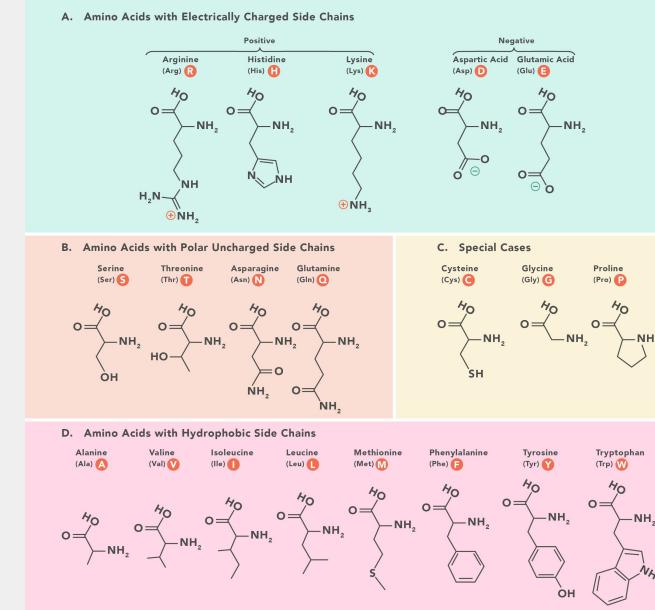
m = ['>', '<', '^', 'v', 'o']

cmap_dict = {'Polar': 'r',
             'Special': 'y',
             'Positive': 'g',
             'Negative': 'g',
             'Hydrophobic': 'm'}

m_dict = {'Positive': '^',
          'Negative': 'v',
          'Polar': '>',
          'Special': '<',
          'Hydrophobic': 'o'}
```

# RESULTS

## PROOF OF CONCEPT



```
# query each amino acid from loaded word2vec model
aa_vec = []
for i in aa:
    aa_vec.append(word_vectors[i])
aa_vec = np.array(aa_vec)

# reduce dimensions of the 200 dimension vector
umap_embed = umap.UMAP(n_neighbors=3, min_dist=0.01, random_state=42).fit_transform(aa_vec)
df = pd.DataFrame([aa, aa_properties, aa_short, umap_embed[:,0], umap_embed[:,1]]).T

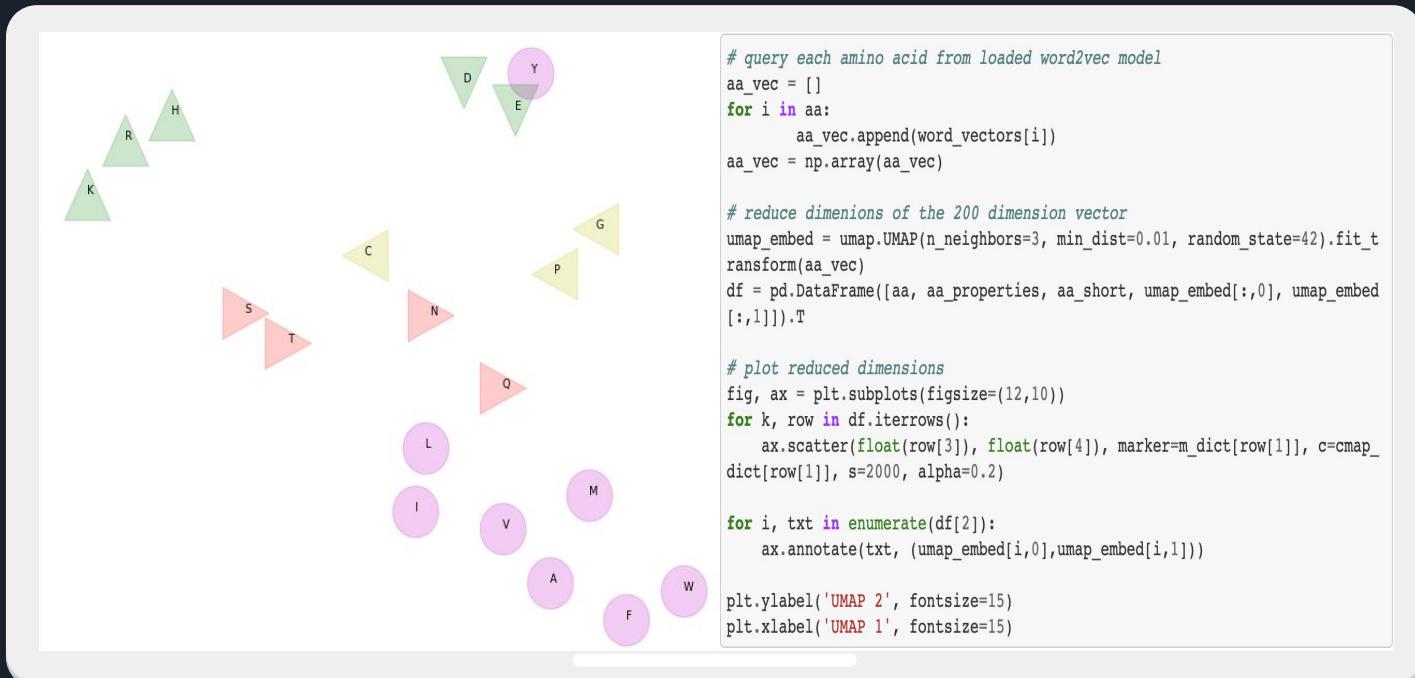
# plot reduced dimensions
fig, ax = plt.subplots(figsize=(12,10))
for k, row in df.iterrows():
    ax.scatter(float(row[3]), float(row[4]), marker=m_dict[row[1]], c=cmap_dict[row[1]], s=2000, alpha=0.2)

for i, txt in enumerate(df[2]):
    ax.annotate(txt, (umap_embed[i,0],umap_embed[i,1]))

plt.ylabel('UMAP 2', fontsize=15)
plt.xlabel('UMAP 1', fontsize=15)
```

# RESULTS

## PROOF OF CONCEPT





# LOADING Libraries scRNA-seq data gene2vec scores

# RESULTS DEMONSTRATION

```
# load libraries
import pickle
import umap
import umap.plot
import
matplotlib.pyplot as
plt
import numpy as np
import os
import scipy.io
import scanpy as sc
from scipy import
sparse
import pandas as pd

# import helper utils
for data-analysis
import utils

# load 10x datasets
data_path = '/Users/sjwu/nlp_CellMarkers/garnett/'
expr = scipy.io.mmread(os.path.join(data_path,
"matrix.mtx.gz"))
expr = expr.tocsr()

# load features and barcodes
pdata =
pd.read_csv(os.path.join(data_path, "barcodes.tsv"),
sep='\t')
fdata =
pd.read_csv(os.path.join(data_path, "genes.tsv"),
sep='\t')
N = expr.shape[0]

# downsample our data to avoid bias
idx_sample = utils.downsample(pdata)
expr_ds = expr[:, idx_sample]
pdata_ds = pdata.loc[idx_sample, :]
```



# LOADING Libraries scRNA-seq data gene2vec scores

# RESULTS DEMONSTRATION

```
# Load scores from gene2vec
with open('w2v_pbmc_m2vparams.scores', 'rb') as file:
    data = pickle.load(file)

# format data
data = utils.format_data(data, fdata)

# update keys to match ground_truth cell type label
data_update = {}
data_update['b_cells'] = data['325472302_b_lymphocytes']
data_update['cd14_monocytes'] = data['325470302_monocytes']
data_update['cd34'] = data['cd34']
data_update['cd56_nk'] = data['nk_cell']
data_update['regulatory'] = data['treg']
data = data_update
del data_update

# update fdata with similarity scores
fdata = utils.update_fdata(fdata, data)

# print each datastructure to visualize what it looks like
print(fdata.head())
print(pdata_ds.head())
```

ensembl	gene_id	b_cells	cd14_monocytes	cd34	cd56_nk	regulatory	barcode	celltype
0	ENSG00000243485	MIR1302-10	0.0	0.0	0.0	0.0	CTGTATAACCCAAA_1	b_cells
1	ENSG00000237613	FAM138A	0.0	0.0	0.0	0.0	TCGACCTGTATGGC_1	b_cells
2	ENSG00000186092	OR4F5	0.0	0.0	0.0	0.0	CATCAGGACGGAGA_1	b_cells
3	ENSG00000238009	RP11-34P13.7	0.0	0.0	0.0	0.0	ACTCAGGATCGCAA_1	b_cells
4	ENSG00000239945	RP11-34P13.8	0.0	0.0	0.0	0.0	CTGTGAGAGGTTG_1	b_cells



## CLASSIFYING INDIVIDUAL CELLS

# RESULTS DEMONSTRATION

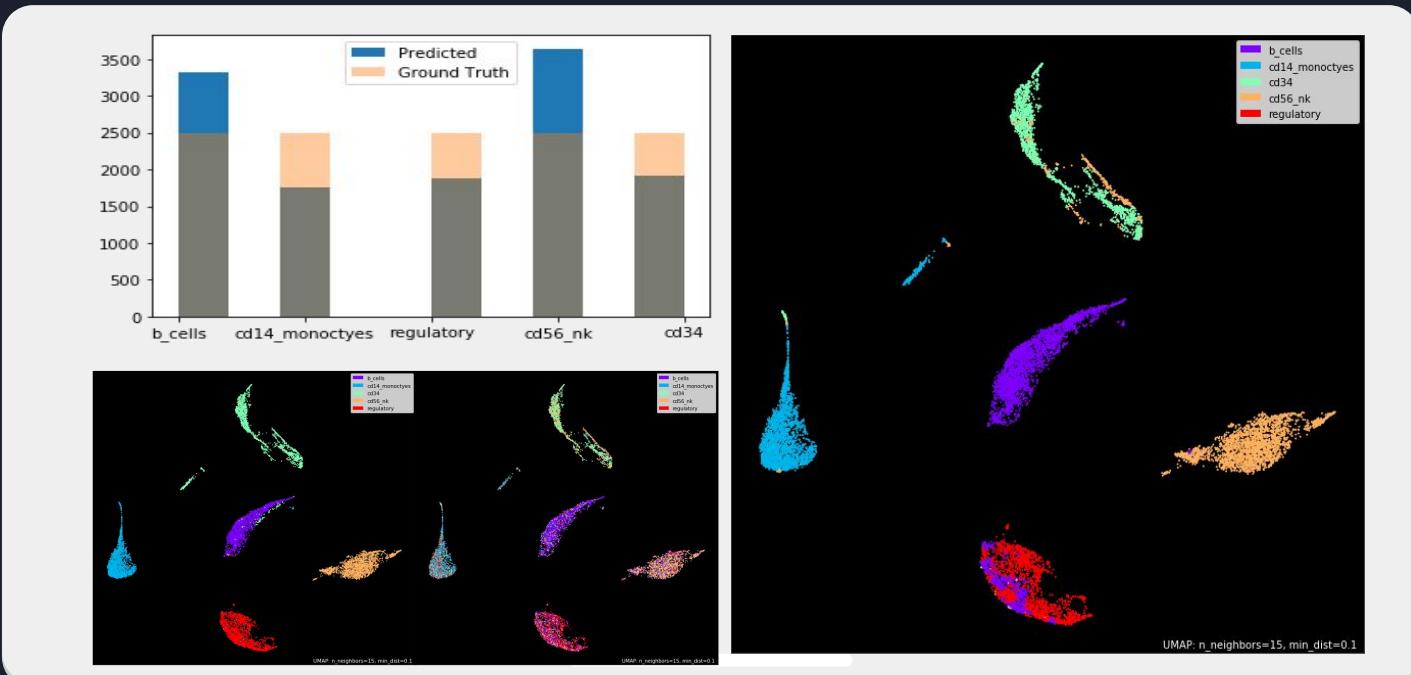
```
# generate a dictionary to map celltypes to index assignments
celltypes = ['b_cells', 'cd14_monomocytes', 'cd34', 'cd56_nk', 'regulatory']
ct_map = dict(zip([i for i in range(len(celltypes))], celltypes))

# normalize our scores from gene2vec
# Upweight higher ranked genes by taking the exponential of the scores.
arr_data = np.array(adata.var[celltypes])
arr_data = (1e5**arr_data)-1
arr_data = arr_data/np.sum(arr_data, axis=0)*1000
arr_data_sprse = sparse.csr_matrix(arr_data)

# calculate scores by dot product
scores = tfidf.dot(arr_data_sprse).argmax(axis=1)
scores = np.array(scores).squeeze()

# compute accuracy
acc = utils.accuracy_score([ct_map[i] for i in scores], adata.obs['celltype'])
print("A naive classification approach of single cells from scRNA-seq data yields:", acc)
```

# RESULTS DEMONSTRATION



VISUALISATION

Thank you!

