# Visualizing representations of problems and skills

**Yifat Amir, Allen Guo, Sam Lau**

In [57]:
```
import pandas as pd
import numpy as np
import json
import seaborn as sns
%matplotlib inline
import matplotlib.pyplot as plt
sns.set()
sns.set_context('talk')
```

In [6]:
```
skill_df = pd.read_csv('skill.tsv', sep='\t').drop('Unnamed: 0', axis=1)
skill_dict = {}
with open('skill_dict.json', 'r', encoding='utf-8') as f:
    loaded = json.load(f)
    for k, v in loaded.items():
        skill_dict[k] = int(v)

skill_num = len(skill_dict) + 1 # including 0
skill_num
```

Out[6]: 111

In [7]:
```
skill_df.head()
```

Out[7]:

| | student_id | s0 | s1 | s2 | s3 | s4 | s5 | s6 | s7 | s8 | ... | s90 | s91 | s92 | s93 | s94 | s95 | s96 | s97 | s98 | s99 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 64525 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | ... | 15 | 15 | 15 | 15 | 16 | 16 | 17 | 18 | 18 | 18 |
| **1** | 70363 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | ... | 8 | 8 | 8 | 8 | 8 | 9 | 9 | 9 | 9 | 9 |
| **2** | 70677 | 1 | 1 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | ... | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 15 |
| **3** | 70695 | 1 | 1 | 3 | 7 | 7 | 7 | 7 | 7 | 7 | ... | 33 | 33 | 33 | 33 | 33 | 33 | 33 | 33 | 33 | 33 |
| **4** | 70699 | 1 | 1 | 2 | 6 | 6 | 6 | 6 | 6 | 6 | ... | 42 | 42 | 42 | 44 | 44 | 44 | 45 | 46 | 46 | 46 |

5 rows × 101 columns

In [44]:
```
sentences = (skill_df
 .iloc[:, 1:]
 .stack()
 .reset_index()
 .rename(columns={'level_0': 'student_id', 0: 'skill'})
 .drop('level_1', axis=1)
 .astype({'skill': str})
 .groupby('student_id')
 .agg(lambda skills: list(skills))
)
sentences.head()
```

Out[44]:

| student_id | skill |
|---|---|
| 0 | [1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, ... |
| 1 | [1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 4, 4, ... |
| 2 | [1, 1, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, ... |
| 3 | [1, 1, 3, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 8, 8, ... |
| 4 | [1, 1, 2, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, ... |

## Question 1: Training

In [46]:
```
from gensim.models.word2vec import Word2Vec

params = {
    'size': 20,
    'window': 10,
    'min_count': 10,
    'iter': 30,
}

model = Word2Vec(sentences=sentences['skill'], **params)
model
```

Out[46]: <gensim.models.word2vec.Word2Vec at 0x11ebe0358>

In [28]:
```
model
```

Out[28]:
```
array([-0.76279932,  2.07639027, -0.11014464,  1.75695968, -1.60743272,
       -3.45588779,  1.10591328, -0.96971965,  3.56253886,  0.13564005,
        1.29442453,  3.53213596, -2.52763629, -1.42959082,  5.8403554 ,
       -2.68304086, -1.71696377,  2.94867325, -1.92379141, -4.78013945], dtype=float32)
```

## Question 2: Dimensionality reduction

In [52]:
```
from sklearn.manifold import TSNE

model.wv.syn0.shape
```

Out[52]: (80, 20)

In [53]:
```
X = model.wv.syn0
X_trans = TSNE(n_components=2).fit_transform(X)
X_trans.shape
```

Out[53]: (80, 2)

In [80]:
```
sk_id_df = (pd.DataFrame(list(skill_dict.items()), columns=['name', 'skill_id'])
            .astype({'skill_id': str})
            .set_index('skill_id')
           )
sk_id_df.head()
```

Out[80]:

| skill_id | name |
|---|---|
| 86 | Order of Operations All |
| 108 | Simplifying Expressions positive exponents |
| 27 | Perimeter of a Polygon |
| 14 | Probability of Two Distinct Events |
| 9 | Mean |

In [82]:
```
export = pd.DataFrame(X_trans, index=model.wv.index2word, columns=['x', 'y']).join(sk_id_df)
export.head()
```

Out[82]:

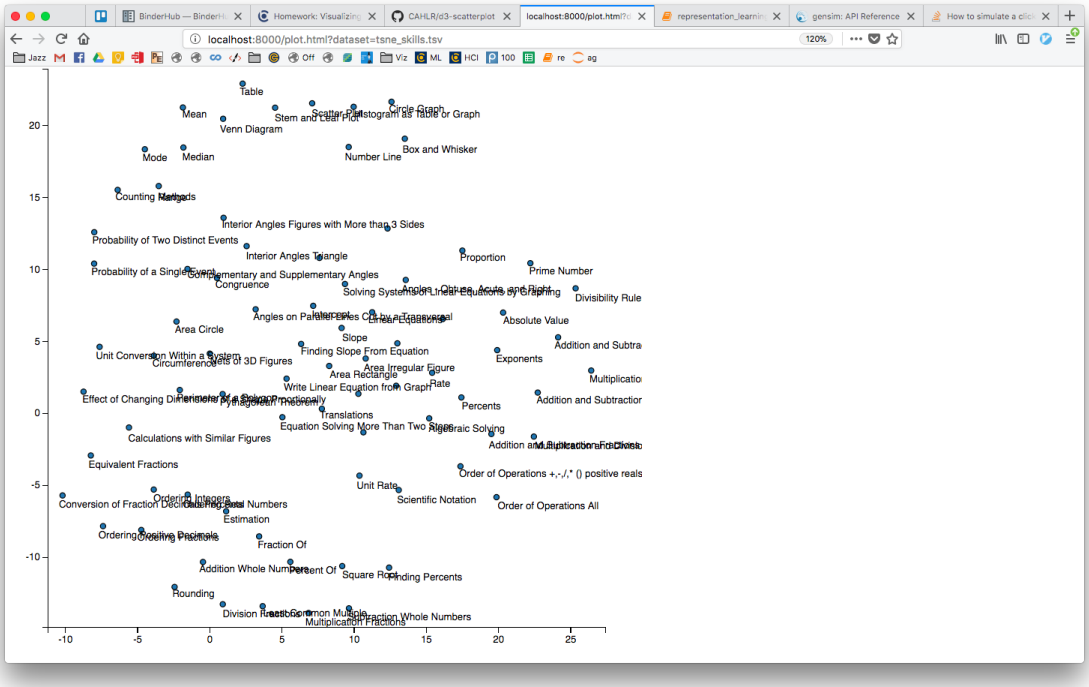
| | x | y | name |
|---|---|---|---|
| 7 | 2.272880 | 22.902727 | Table |
| 30 | -10.199936 | -5.702681 | Conversion of Fraction Decimals Percents |
| 8 | 0.922563 | 20.461325 | Venn Diagram |
| 2 | 12.585614 | 21.643568 | Circle Graph |
| 33 | -4.746242 | -8.102057 | Ordering Fractions |

In [83]:
```
export.to_csv('tsne_skills.tsv', sep='\t', index=False)
```

In [86]:
```
!head tsne_skills.tsv
!cp tsne_skills.tsv ../../d3-scatterplot/
```

```
x	y	name
2.2728798389434814	22.902727127075195	Table
-10.199935913085938	-5.702680587768555	Conversion of Fraction Decimals Percents
0.9225633144378662	20.46132469177246	Venn Diagram
12.585614204406738	21.64356803894043	Circle Graph
-4.746241569519043	-8.102057456970215	Ordering Fractions
4.523916244506836	21.230213165283203	Stem and Leaf Plot
-7.395717620849609	-7.834371566772461	Ordering Positive Decimals
-1.8299428224563599	18.463851928710938	Median
13.507652282714844	19.07671356201172	Box and Whisker
```

### d3-scatterplot Visualization:




## Question 3

We notice that similar skills are close together in the visualization. For example, Mode is immediately next to Mean and Median, and Slope is next to Finding Slope From Equation.

It also looks like skills that are far away are less related. For example, Order of Operations All is far from Square Root on the plot and these skills are less related. This implies that few students did both Order of Operations and Square Roots in the same problem sequence.

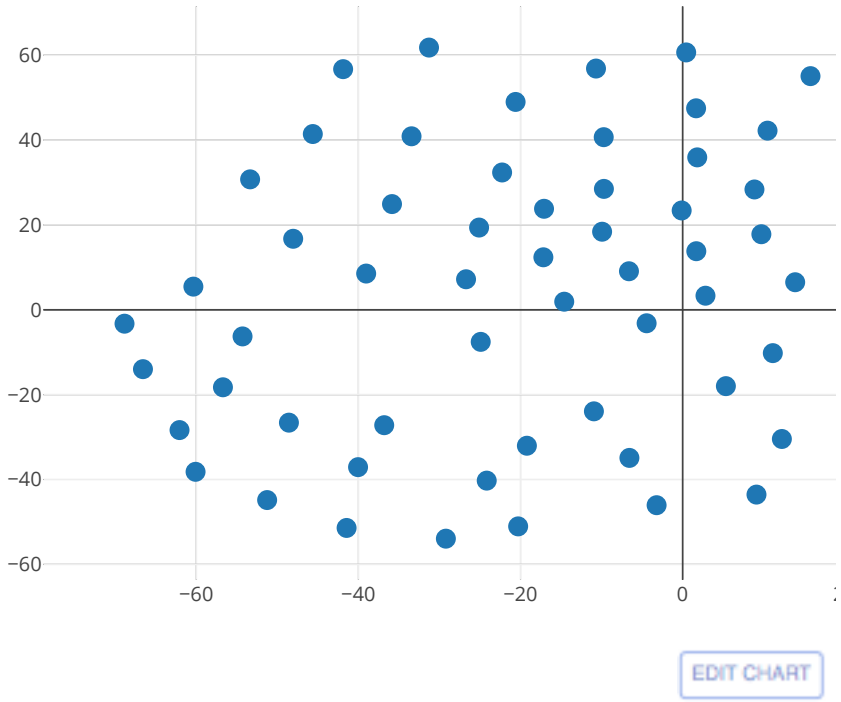
The proximity of skills can be interpreted as how often these skills were practiced by students sequentially.

## Question 4

In [132]:
```
import plotly
import plotly.plotly as py
import plotly.graph_objs as go

def train_reduce(sentences, params):
    model = Word2Vec(sentences=sentences, **params)

    X = model.wv.syn0
    X_trans = TSNE(n_components=2).fit_transform(X)
    tsne_model = pd.DataFrame(X_trans, index=model.wv.index2word, columns=['x', 'y']).join(s
    return tsne_model

def tsne_plot(name, points, color=None):
    if color is None:
        color = 0

    trace = go.Scatter(
        x = points['x'],
        y = points['y'],
        text = points['name'],
        mode = 'markers',
        marker=dict(
            size='12',
            color = color,
            colorscale='Viridis',
            showscale=True
        )
    )
    data = [trace]

    return py.iplot(data, filename=name)

def train_reduce_plot(name, sentences, params, color=None):
    return tsne_plot(name, train_reduce(sentences, params), color)
```

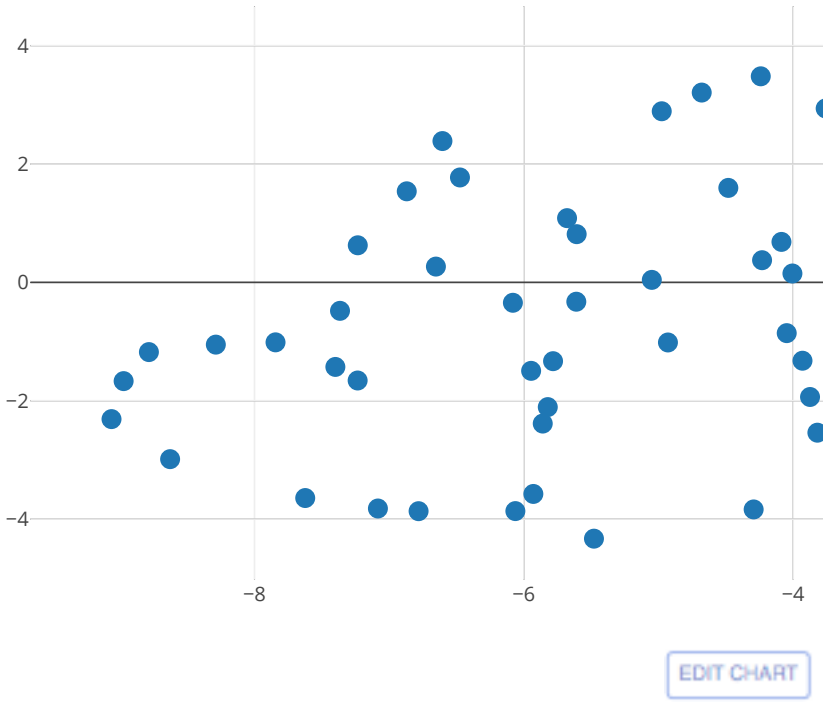
In [133]:
```
train_reduce_plot('default', sentences['skill'], {
    'size': 20,
    'window': 10,
    'min_count': 10,
    'iter': 30,
})
```

Out[133]:



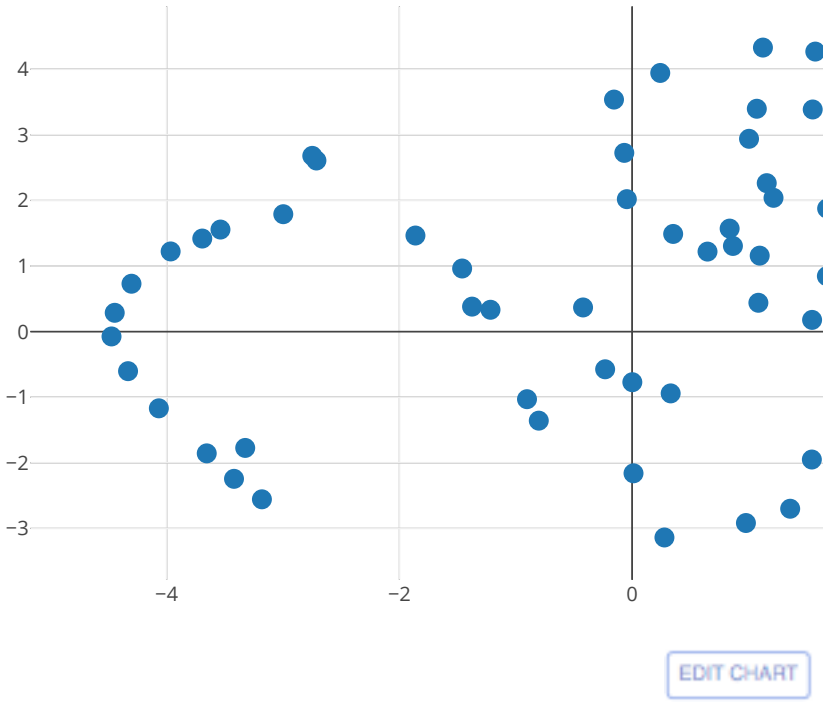
In [128]:
```
train_reduce_plot('window=1', sentences['skill'], {
    'size': 20,
    'window': 1,
    'min_count': 10,
    'iter': 30,
})
```

Out[128]:



In [129]:
```
train_reduce_plot('size=50', sentences['skill'], {
    'size': 50,
    'window': 10,
    'min_count': 10,
    'iter': 30,
})
```

Out[129]:




Smaller window size appears to create denser clusters. This might be because Word2Vec is more certain about skill representations when there are less skills as context.

As far we could tell, vector size doesn't seem to have a noticeable effect on the TSNE visualization scatterplot.

## Question 5

In [111]:
```
assistment_df = pd.read_csv('assistment_id.tsv', sep='\t').drop('Unnamed: 0', axis=1)
assistment_df.head()
```

Out[111]:

| | student_id | a0 | a1 | a2 | a3 | a4 | a5 | a6 | a7 | a8 | ... | a90 | a91 | a92 | a93 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 64525 | 33139 | 33150 | 52640 | 52655 | 52647 | 53476 | 53477 | 53458 | 36836 | ... | 41846 | 41862 | 37038 | 37027 |
| **1** | 70363 | 33110 | 33172 | 33174 | 33123 | 31398 | 36845 | 31412 | 36841 | 36821 | ... | 31322 | 31313 | 31326 | 31328 |
| **2** | 70677 | 33168 | 33112 | 31968 | 31970 | 53345 | 53301 | 53317 | 31960 | 36433 | ... | 36517 | 36521 | 36541 | 36522 |
| **3** | 70695 | 33149 | 33188 | 33040 | 32023 | 32017 | 32021 | 32054 | 36479 | 36463 | ... | 55641 | 55646 | 55632 | 55659 |
| **4** | 70699 | 33145 | 33151 | 53417 | 36901 | 36938 | 53592 | 53579 | 53617 | 53612 | ... | 34592 | 34606 | 34605 | 53417 |

5 rows × 101 columns

In [116]:
```
assist_sentences = (assistment_df
 .iloc[:, 1:]
 .stack()
 .reset_index()
 .rename(columns={'level_0': 'student_id', 0: 'prob_id'})
 .drop('level_1', axis=1)
 .astype({'prob_id': str})
 .groupby('student_id')
 .agg(lambda skills: list(skills))
)
assist_sentences.head()
```

Out[116]:

| | prob_id |
|---|---|
| **student_id** | |
| **0** | [33139, 33150, 52640, 52655, 52647, 53476, 534... |
| **1** | [33110, 33172, 33174, 33123, 31398, 36845, 314... |
| **2** | [33168, 33112, 31968, 31970, 53345, 53301, 533... |
| **3** | [33149, 33188, 33040, 32023, 32017, 32021, 320... |
| **4** | [33145, 33151, 53417, 36901, 36938, 53592, 535... |

In [135]:
```
%%time

assist_points = train_reduce(assist_sentences['prob_id'], {
    'size': 20,
    'window': 10,
    'min_count': 10,
    'iter': 30,
})
assist_points.head()
```

Out[135]:

| | x | y | name |
|---|---|---|---|
| **36467** | -19.226925 | -13.266063 | NaN |
| **36488** | -18.889935 | -12.955877 | NaN |
| **36768** | -67.145325 | -5.832832 | NaN |
| **33107** | 21.221672 | 60.954773 | NaN |
| **31308** | -61.107166 | 3.914618 | NaN |

In [151]:
```
skill_seq = (skill_df
 .iloc[:, 1:]
 .stack()
 .reset_index(drop=True)
 .rename('skill_id')
)
skill_seq.head()
```

Out[151]:
```
0    1
1    1
2    2
3    2
4    2
Name: skill_id, dtype: int64
```

In [158]:
```
assist_to_skill = (assistment_df
 .iloc[:, 1:]
 .stack()
 .reset_index(drop=True)
 .rename('prob_id')
 .to_frame()
 .join(skill_seq)
 .groupby('prob_id')
 .first()
)
assist_to_skill.head()
```

Out[158]:

| prob_id | skill_id |
|---|---|
| **86** | 31 |
| **252** | 31 |
| **810** | 44 |
| **849** | 47 |
| **1080** | 44 |

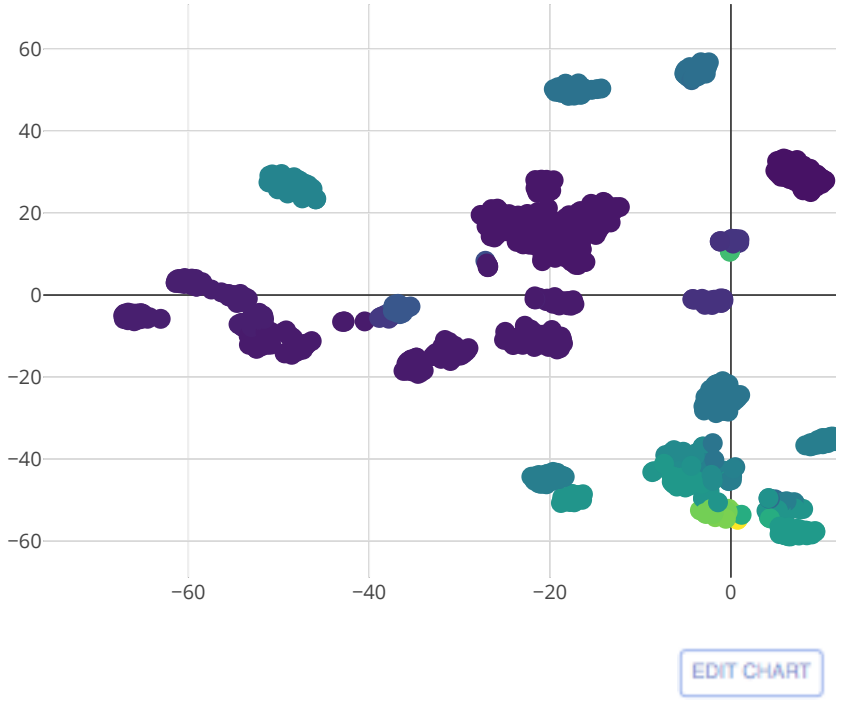
In [175]:
```
assist_points.index = assist_points.index.astype('int')
assist_points_with_skill = assist_points.join(assist_to_skill)
assist_points_with_skill['name'] = assist_points_with_skill['skill_id']
assist_points_with_skill.head()
```

Out[175]:

| | x | y | name | skill_id |
|---|---|---|---|---|
| **36467** | -19.226925 | -13.266063 | 7 | 7 |
| **36488** | -18.889935 | -12.955877 | 7 | 7 |
| **36768** | -67.145325 | -5.832832 | 8 | 8 |
| **33107** | 21.221672 | 60.954773 | 1 | 1 |
| **31308** | -61.107166 | 3.914618 | 8 | 8 |

In [176]:
```
tsne_plot('assist_with_skill', assist_points_with_skill, assist_points_with_skill['skill_id'
```

Out[176]:




We notice strong clusters of assistment questions by the skills that they evaluate!