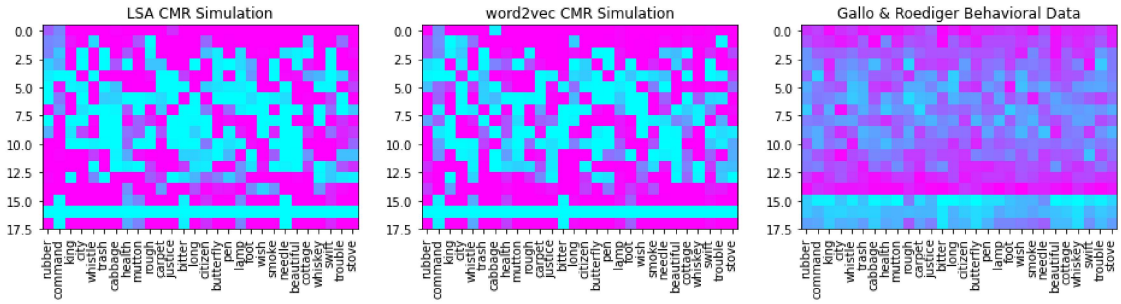
1.2 Word2vec Simulation Results

[23]:
```
# Load up our matlab simulation output similar to LSA procedure
cmr_w2v_results = dict()
loadmat('cmr_w2v_drm.mat',cmr_w2v_results,mat_dtype=True)
cmr_w2v_list_data = cmr_w2v_results['data'][0][0][0][0]
cmr_w2v_list_data = {w: cmr_w2v_list_data[i][0].transpose() for i, w in␣
 ↪enumerate(list_order)}
# We now have for our cmr simulated free recall tasks, subject responses to␣
 ↪list item recall (in order)
# per column stored as a matrix in each key of dictionary (corresponding to␣
 ↪list critical item)
# A '-1' stands for critical lure recall
print(cmr_w2v_list_data['rubber'])
# Save a formatted .csv with w2v semantic matrix simulation
save_matlab_output('raw_cmr_w2v_data.csv',cmr_w2v_list_data)
# Simulated CMR (w2v) Data
cmr_w2v_subjects, cmr_w2v_list_order, cmr_w2v_word_lists, cmr_w2v_recall_data,␣
 ↪cmr_w2v_list_data, cmr_w2v_subject_matrix, cmr_w2v_list_matrix =␣
 ↪load_data("raw_cmr_w2v_data.csv")
```

```
[[14. 14.  1. … 14. 15. 15.]
 [-1. 15. 13. … 15. 14. 14.]
 [12.  2. 11. … 13.  1.  1.]
 …
 [ 0.  0.  0. …  9.  0.  0.]
 [ 0.  0.  0. …  0.  0.  0.]
 [ 0.  0.  0. …  0.  0.  0.]]
```

[24]:
```
# Matrix heatmap for all lists
f,axes = plt.subplots(1,3,figsize=(16,4))
plt.sca(axes[0])
axes[0].title.set_text("LSA CMR Simulation")
plt.xticks(ticks=range(len(cmr_list_order)),labels=cmr_list_order,rotation=90)
plt.
 ↪imshow(format_recall_matrix(cmr_list_matrix,cmr_list_order),cmap='cool',interpolation='near
plt.sca(axes[1])
axes[1].title.set_text("word2vec CMR Simulation")
plt.
 ↪xticks(ticks=range(len(cmr_w2v_list_order)),labels=cmr_w2v_list_order,rotation=90)
plt.
 ↪imshow(format_recall_matrix(cmr_w2v_list_matrix,cmr_w2v_list_order),cmap='cool',interpolati
plt.sca(axes[2])
axes[2].title.set_text("Gallo & Roediger Behavioral Data")
```

```
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.
 ↪imshow(format_recall_matrix(list_matrix,list_order),cmap='cool',interpolation='nearest')
plt.show()
```



1.2.1 Preliminary Analysis

We observe that the LSA and word2vec results are comparable in its deterministic nature of recall simulation. Although word2vec appears to generate greater frequency of recall. Furthermore, a greater proportion of lists using word2vec semantic matrix exhibits false recall as compared to using LSA semantic matrix.

In addition, the CMR simulation results appears to be highly deterministic and show little variation among simulated trials (as expected since only small pertubations are added as random noise in accumulation step) unlike behavioral results. In the simulation, many terms are either fully (100%) recalled in all trials or not recalled at all.

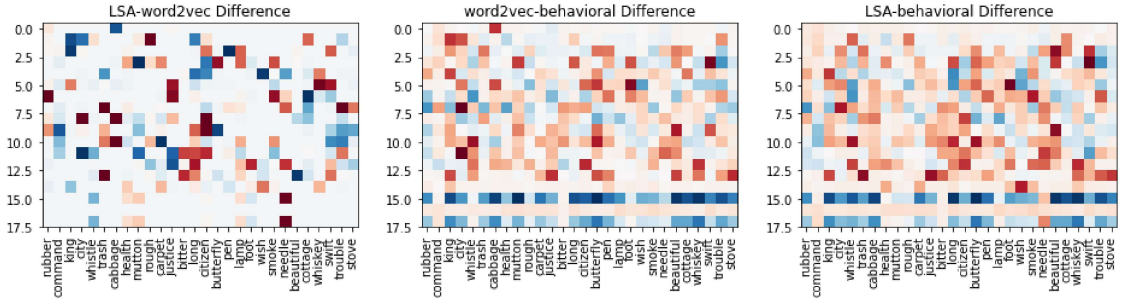
In the next section, we confirm our suspicions that the semantic force of word connections is what's driving false recall and recall of list items.

1.2.2 Side-track: What's the squared-error between the diagrams above?

Below, we plot the squared-error matrix between LSA-word2vec, LSA-behavioral, and word2vec-behavioral.

[25]:
```
# Matrices for LSA,word2vec,behavioral data
lsa_mat = format_recall_matrix(cmr_list_matrix,cmr_list_order)
w2v_mat = format_recall_matrix(cmr_w2v_list_matrix,cmr_w2v_list_order)
behavioral_mat = format_recall_matrix(list_matrix,list_order)

f,axes = plt.subplots(1,3,figsize=(16,4))
# LSA-word2vec squared difference (preserving sign)
plt.sca(axes[0])
axes[0].title.set_text("LSA-word2vec Difference")
plt.xticks(ticks=range(len(cmr_list_order)),labels=list_order,rotation=90)
diff_mat = lsa_mat-w2v_mat
```

```
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# word2vec-behavioral squared difference
plt.sca(axes[1])
axes[1].title.set_text("word2vec-behavioral Difference")
plt.xticks(ticks=range(len(cmr_w2v_list_order)),labels=list_order,rotation=90)
diff_mat = w2v_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# LSA-behavioral squared difference
plt.sca(axes[2])
axes[2].title.set_text("LSA-behavioral Difference")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
diff_mat = lsa_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
plt.show()
```



In the above plots, we use a Red-Blue color map with the following value mapping:

- Intensity of colors represent magnitude of values
- <0 Negative values are represented by **Red**
- >0 Positive values are represented by **Blue**
- ~0 Values close to 0 are lightly shaded, close to **White**

Each plot is labeled with the direction of subtraction (for instance last plot is generated by subtracting values in the bahavioral matrix *from* the LSA matrix).

As we can see above, the difference in simulated results between using LSA and word2vec are not very different, however, in regions where they are different, perhaps this difference should be in cosine values within lists for those items (which we will confirm in the next section). Furthermore, comparing either set of simulation data against the behavioral results, we see that there are two regions of difference. - The first region is mostly distributed around the middle of the list where the CMR simulation consistently **under-recalls list items near the middle**. - The second region is the region of false recalls (last 3 rows) where the CMR simulation consistently **over-recalls critical lure items**

At this point, before dismissing CMR simulation as inaccurate, we should note that the parameters

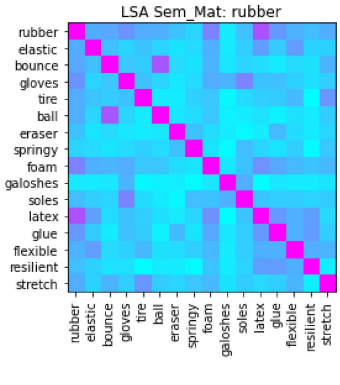
used for this simulation are those which have been determined to fit random lists drawn from the Murdock,1962 set. A fair comparison would require us to run the same parameter fit on these 28 lists instead using the reported Genetic Algorithm in Poly et.al 2009.

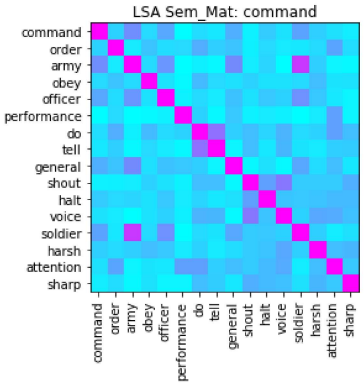
1.3 Analysis of semantic structure within lists

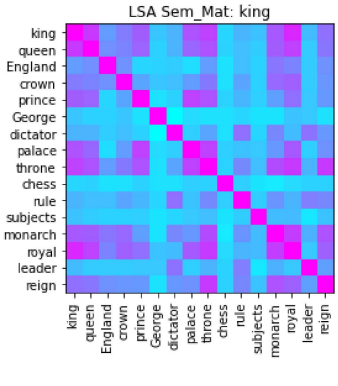
First, we take a look at both the LSA and word2vec semantic connectivity within list items and compute the correlation of recall probability to overall strength of connection with other list items. To get an initial sense of how interconnected are the lists, we simply plot the matrix of cosines for each list.

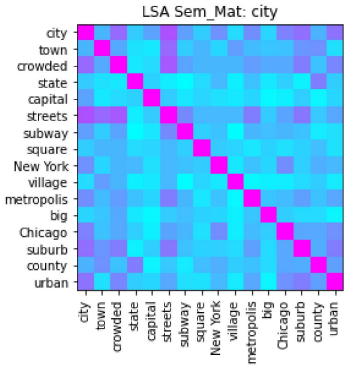
[26]:
```
# Generate matrices for cosines within each list
list_w2v_mat = {crit: np.zeros((len(word_lists[crit])+1,␣
 ↪len(word_lists[crit])+1)) for crit in list_order}
list_lsa_mat = {crit: np.zeros((len(word_lists[crit])+1,␣
 ↪len(word_lists[crit])+1)) for crit in list_order}

txy = 0 #To track where in diagonal matrix our local list matrix starts
for crit in list_order:
    etxy = txy+len(word_lists[crit])+1
    list_lsa_mat[crit] = sem_mat[txy:etxy,txy:etxy]
    list_w2v_mat[crit] = w2v_sem_mat[txy:etxy,txy:etxy]
    txy = etxy
```

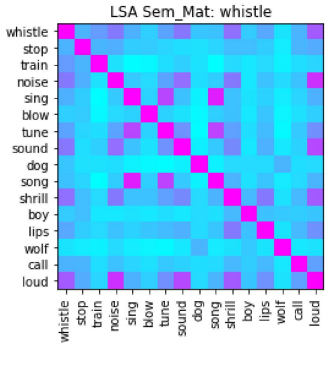
[27]:
```
# Plot for LSA Semantic Matrices
num_col = 3
x = 0
for i, crit in enumerate(list_order):
    x = i%num_col
    if (i%num_col == 0):
        plt.show()
        f,axes = plt.subplots(1,num_col,figsize=(16,4))
    cur_list_words = [crit] + word_lists[crit]
    plt.sca(axes[x])
    plt.
 ↪xticks(ticks=range(len(cur_list_words)),labels=cur_list_words,rotation=90)
    plt.yticks(ticks=range(len(cur_list_words)),labels=cur_list_words)
    axes[x].title.set_text(f"LSA Sem_Mat: {crit}")
    plt.imshow(list_lsa_mat[crit],cmap='cool',interpolation='nearest')
plt.show()
```

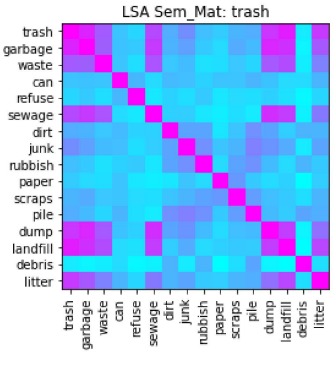

LSA Sem_Mat: rubber
rubber
elastic
bounce
gloves
tire
ball
eraser
springy
foam
galoshes
soles
latex
glue
flexible
resilient
stretch

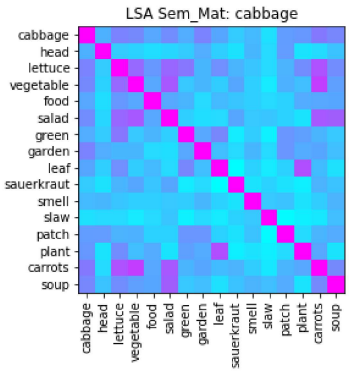

LSA Sem_Mat: command
command
order
army
obey
officer
performance
do
tell
general
shout
halt
voice
soldier
harsh
attention
sharp

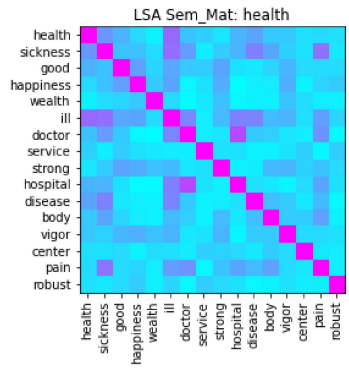

LSA Sem_Mat: king
king
queen
England
crown
prince
George
dictator
palace
throne
chess
rule
subjects
monarch
royal
leader
reign

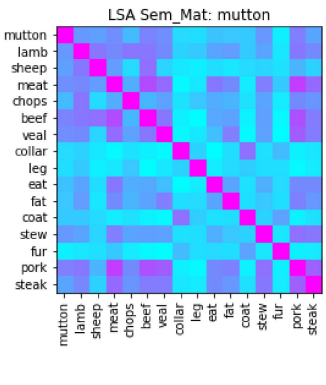

LSA Sem_Mat: city
city
town
crowded
state
capital
streets
subway
square
New York
village
metropolis
big
Chicago
suburb
county
urban

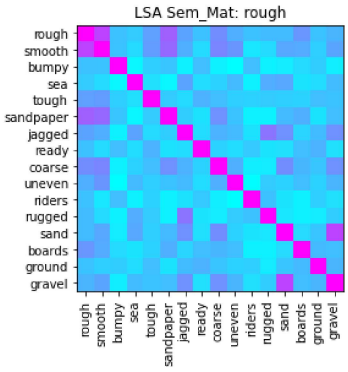

LSA Sem_Mat: whistle
whistle
stop
train
noise
sing
blow
tune
sound
dog
song
shrill
boy
lips
wolf
call
loud

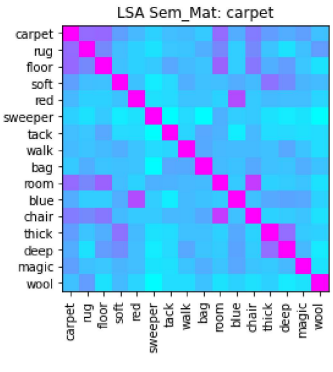

LSA Sem_Mat: trash
trash
garbage
waste
can
refuse
sewage
dirt
junk
rubbish
paper
scraps
pile
dump
landfill
debris
litter

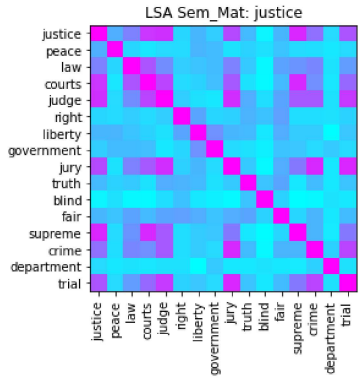

LSA Sem_Mat: cabbage
cabbage
head
lettuce
vegetable
food
salad
green
garden
leaf
sauerkraut
smell
slaw
patch
plant
carrots
soup

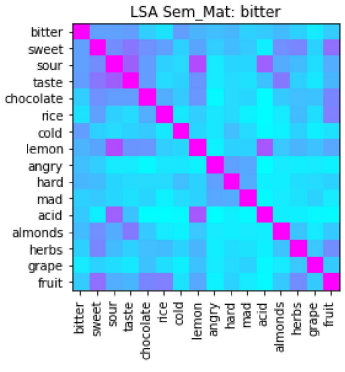

LSA Sem_Mat: health
health
sickness
good
happiness
wealth
ill
doctor
service
strong
hospital
disease
body
vigor
center
pain
robust

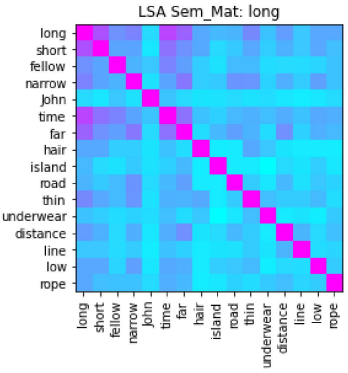

LSA Sem_Mat: mutton
mutton
lamb
sheep
meat
chops
beef
veal
collar
leg
eat
fat
coat
stew
fur
pork
steak

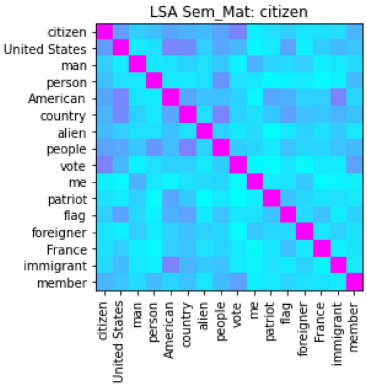

LSA Sem_Mat: rough
rough
smooth
bumpy
sea
tough
sandpaper
jagged
ready
coarse
uneven
riders
rugged
sand
boards
ground
gravel

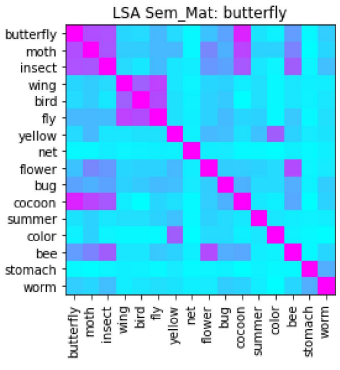

LSA Sem_Mat: carpet
carpet
rug
floor
soft
red
sweeper
tack
walk
bag
room
blue
chair
thick
deep
magic
wool

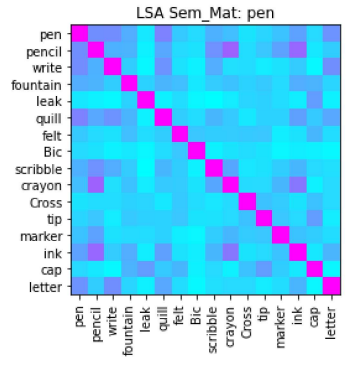

LSA Sem_Mat: justice
justice
peace
law
courts
judge
right
liberty
government
jury
truth
blind
fair
supreme
crime
department
trial

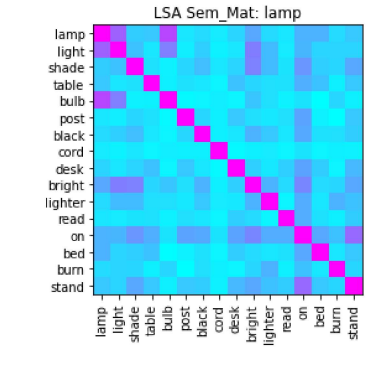

LSA Sem_Mat: bitter
bitter
sweet
sour
taste
chocolate
rice
cold
lemon
angry
hard
mad
acid
almonds
herbs
grape
fruit

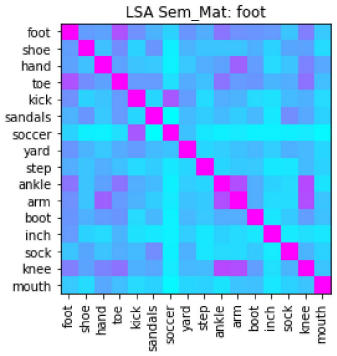

LSA Sem_Mat: long
long
short
fellow
narrow
John
time
far
hair
island
road
thin
underwear
distance
line
low
rope

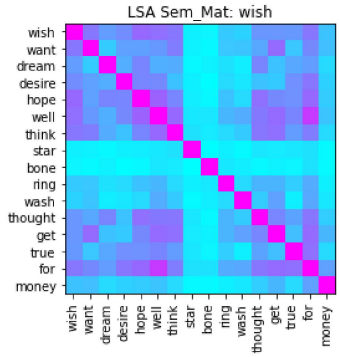

LSA Sem_Mat: citizen
citizen
United States
man
person
American
country
alien
people
vote
me
patriot
flag
foreigner
France
immigrant
member

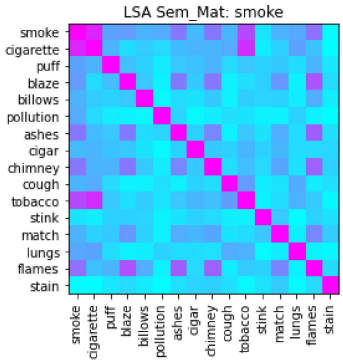

LSA Sem_Mat: butterfly
butterfly
moth
insect
wing
bird
fly
yellow
net
flower
bug
cocoon
summer
color
bee
stomach
worm

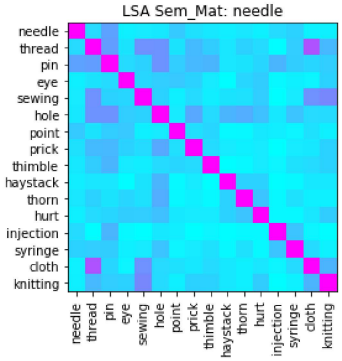

LSA Sem_Mat: pen
pen
pencil
write
fountain
leak
quill
felt
Bic
scribble
crayon
Cross
tip
marker
ink
cap
letter

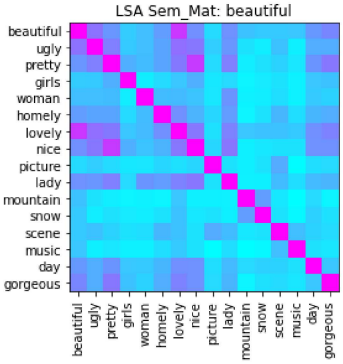

LSA Sem_Mat: lamp
lamp
light
shade
table
bulb
post
black
cord
desk
bright
lighter
read
on
bed
burn
stand

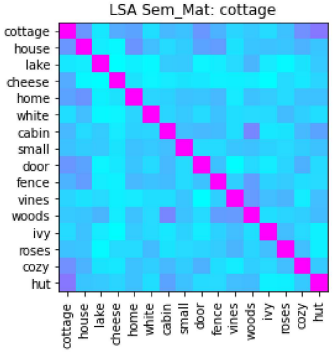

LSA Sem_Mat: foot
foot
shoe
hand
toe
kick
sandals
soccer
yard
step
ankle
arm
boot
inch
sock
knee
mouth


LSA Sem_Mat: wish
wish
want
dream
desire
hope
well
think
star
bone
ring
wash
thought
get
true
for
money

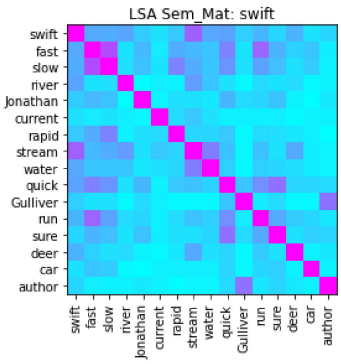

LSA Sem_Mat: smoke
smoke
cigarette
puff
blaze
billows
pollution
ashes
cigar
chimney
cough
tobacco
stink
match
lungs
flames
stain

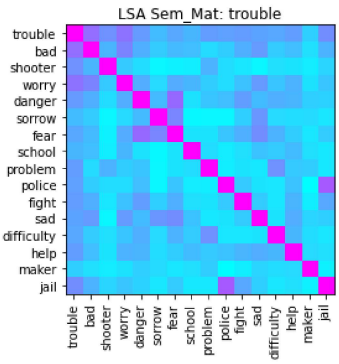

LSA Sem_Mat: needle
needle
thread
pin
eye
sewing
hole
point
prick
thimble
haystack
thorn
hurt
injection
syringe
cloth
knitting

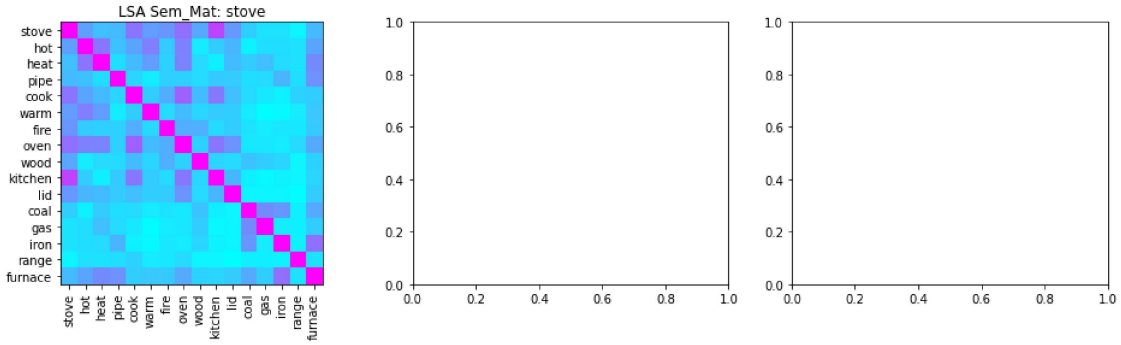

LSA Sem_Mat: beautiful
beautiful
ugly
pretty
girls
woman
homely
lovely
nice
picture
lady
mountain
snow
scene
music
day
gorgeous


LSA Sem_Mat: cottage
cottage
house
lake
cheese
home
white
cabin
small
door
fence
vines
woods
ivy
roses
cozy
hut

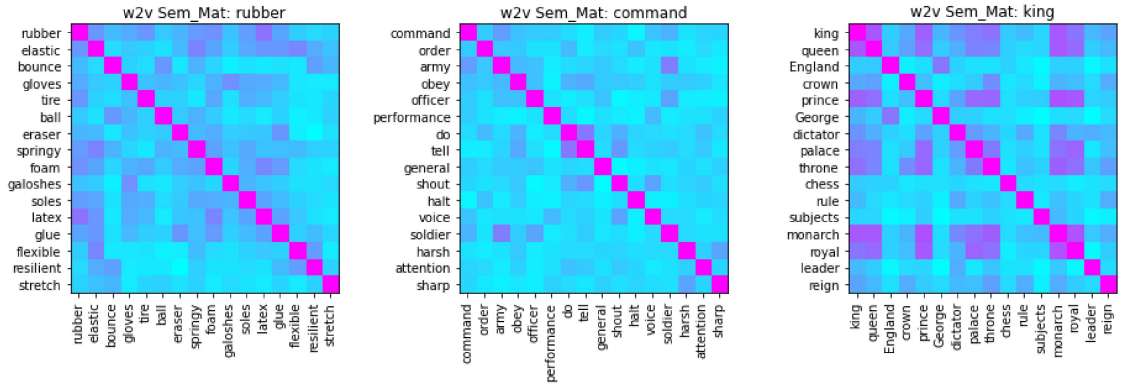

LSA Sem_Mat: whiskey
whiskey
drink
drunk
beer
liquor
gin
bottles
alchohol
rye
glass
wine
rum
bourbon
evil
bar
scotch

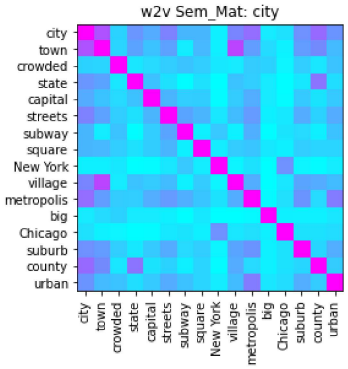

LSA Sem_Mat: swift
swift
fast
slow
river
Jonathan
current
rapid
stream
water
quick
Gulliver
run
sure
deer
car
author

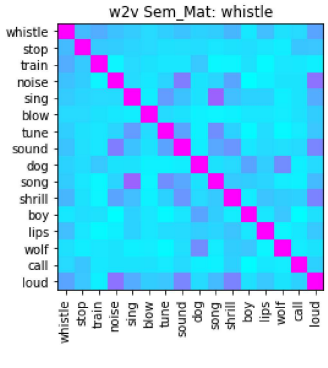

LSA Sem_Mat: trouble
trouble
bad
shooter
worry
danger
sorrow
fear
school
problem
police
fight
sad
difficulty
help
maker
jail

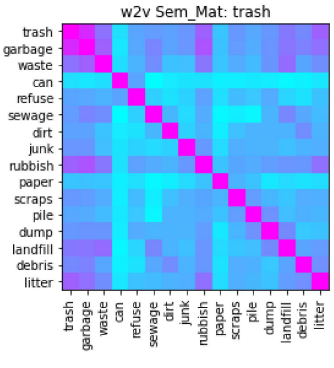


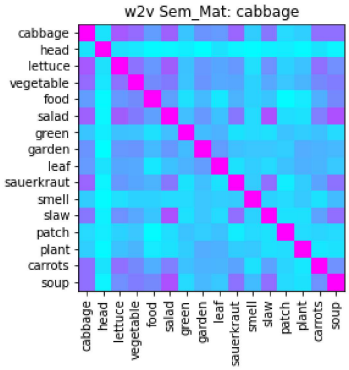
[28]:
```
# Plot for word2vec Semantic Matrices
num_col = 3
x = 0
for i, crit in enumerate(list_order):
    x = i%num_col
    if (i%num_col == 0):
        plt.show()
        f,axes = plt.subplots(1,num_col,figsize=(16,4))
    cur_list_words = [crit] + word_lists[crit]
    plt.sca(axes[x])
    plt.
 ↪xticks(ticks=range(len(cur_list_words)),labels=cur_list_words,rotation=90)
    plt.yticks(ticks=range(len(cur_list_words)),labels=cur_list_words)
    axes[x].title.set_text(f"w2v Sem_Mat: {crit}")
    plt.imshow(list_w2v_mat[crit],cmap='cool',interpolation='nearest')
plt.show()
```

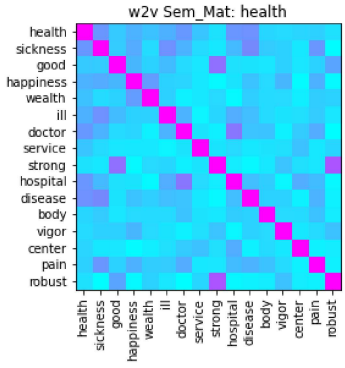


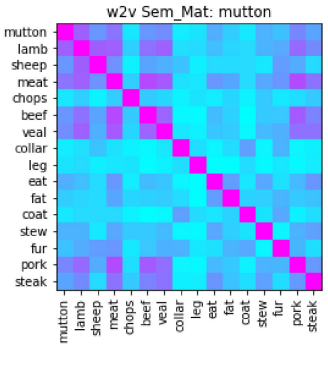

w2v Sem_Mat: city
city
town
crowded
state
capital
streets
subway
square
New York
village
metropolis
big
Chicago
suburb
county
urban

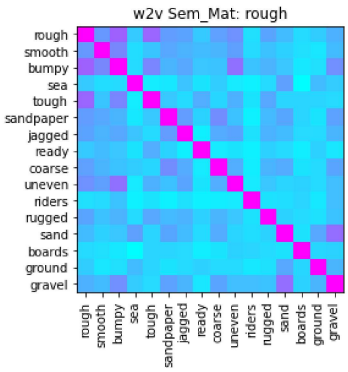

w2v Sem_Mat: whistle
whistle
stop
train
noise
sing
blow
tune
sound
dog
song
shrill
boy
lips
wolf
call
loud

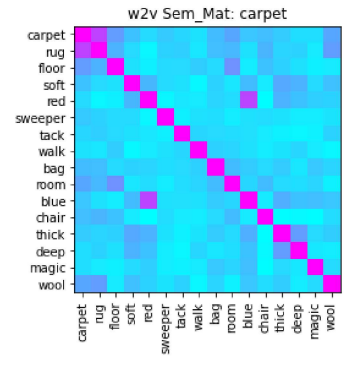

w2v Sem_Mat: trash
trash
garbage
waste
can
refuse
sewage
dirt
junk
rubbish
paper
scraps
pile
dump
landfill
debris
litter

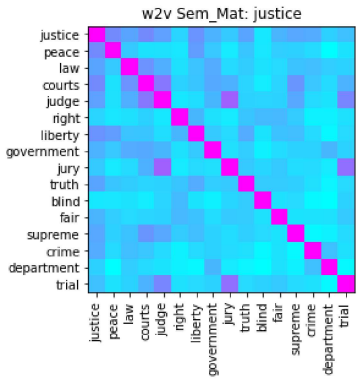

w2v Sem_Mat: cabbage
cabbage
head
lettuce
vegetable
food
salad
green
garden
leaf
sauerkraut
smell
slaw
patch
plant
carrots
soup

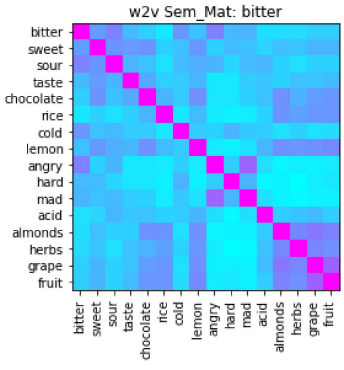

w2v Sem_Mat: health
health
sickness
good
happiness
wealth
ill
doctor
service
strong
hospital
disease
body
vigor
center
pain
robust

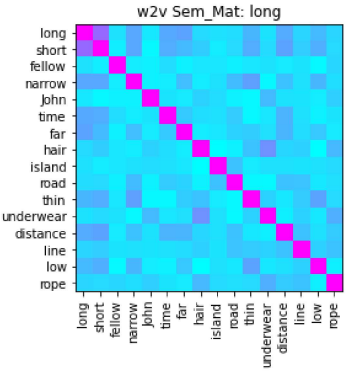

w2v Sem_Mat: mutton
mutton
lamb
sheep
meat
chops
beef
veal
collar
leg
eat
fat
coat
stew
fur
pork
steak

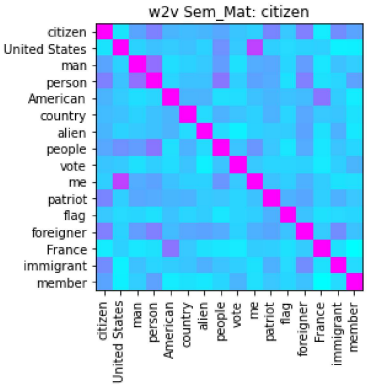

w2v Sem_Mat: rough
rough
smooth
bumpy
sea
tough
sandpaper
jagged
ready
coarse
uneven
riders
rugged
sand
boards
ground
gravel

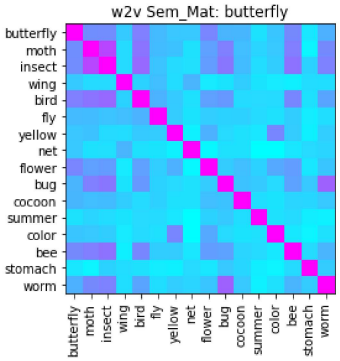

w2v Sem_Mat: carpet
carpet
rug
floor
soft
red
sweeper
tack
walk
bag
room
blue
chair
thick
deep
magic
wool

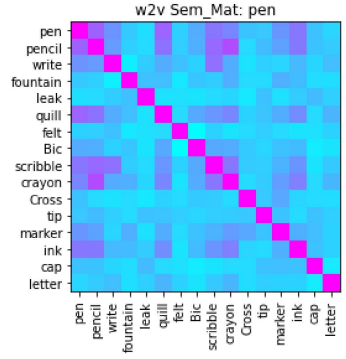

w2v Sem_Mat: justice
justice
peace
law
courts
judge
right
liberty
government
jury
truth
blind
fair
supreme
crime
department
trial

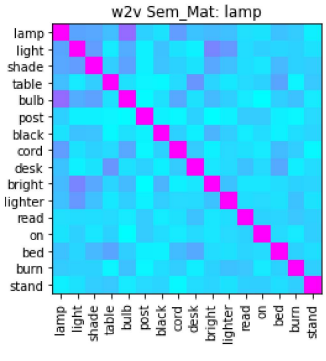

w2v Sem_Mat: bitter
bitter
sweet
sour
taste
chocolate
rice
cold
lemon
angry
hard
mad
acid
almonds
herbs
grape
fruit

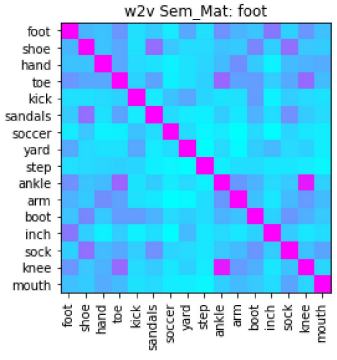

w2v Sem_Mat: long
long
short
fellow
narrow
John
time
far
hair
island
road
thin
underwear
distance
line
low
rope

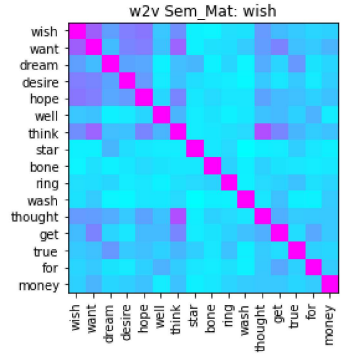

w2v Sem_Mat: citizen
citizen
United States
man
person
American
country
alien
people
vote
me
patriot
flag
foreigner
France
immigrant
member

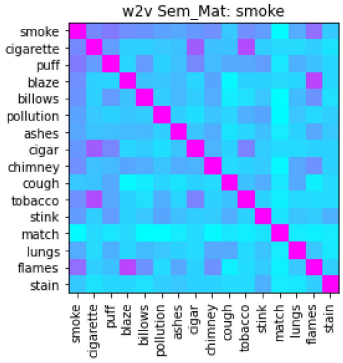

w2v Sem_Mat: butterfly
butterfly
moth
insect
wing
bird
fly
yellow
net
flower
bug
cocoon
summer
color
bee
stomach
worm

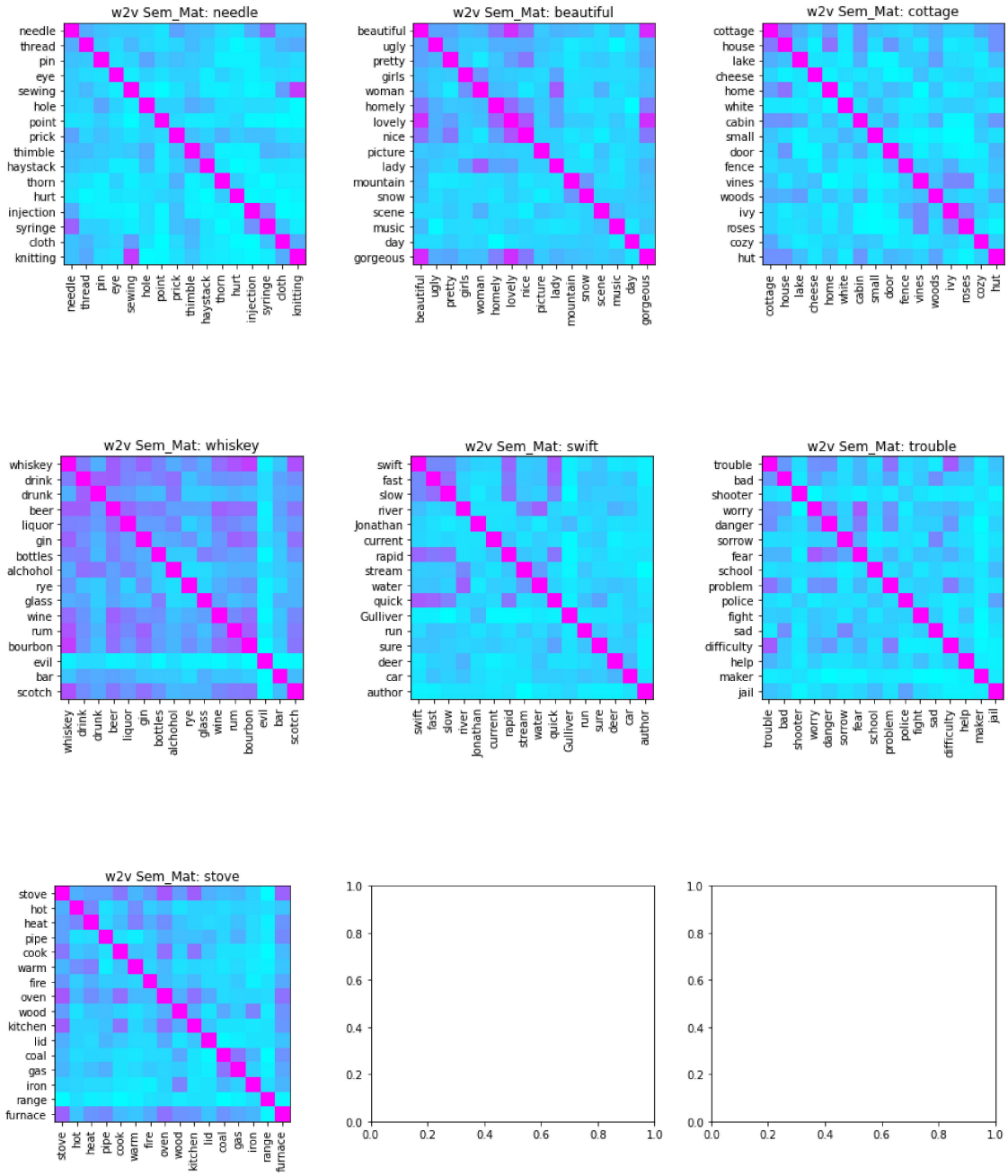

w2v Sem_Mat: pen
pen
pencil
write
fountain
leak
quill
felt
Bic
scribble
crayon
Cross
tip
marker
ink
cap
letter


w2v Sem_Mat: lamp
lamp
light
shade
table
bulb
post
black
cord
desk
bright
lighter
read
on
bed
burn
stand


w2v Sem_Mat: foot
foot
shoe
hand
toe
kick
sandals
soccer
yard
step
ankle
arm
boot
inch
sock
knee
mouth


w2v Sem_Mat: wish
wish
want
dream
desire
hope
well
think
star
bone
ring
wash
thought
get
true
for
money

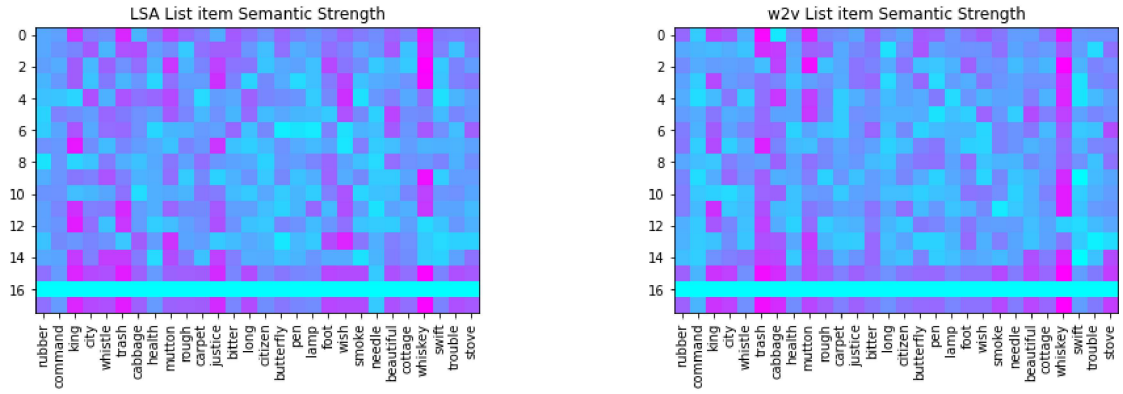

w2v Sem_Mat: smoke
smoke
cigarette
puff
blaze
billows
pollution
ashes
cigar
chimney
cough
tobacco
stink
match
lungs
flames
stain



To test our hyposthesis that the semantic closeness of each item to other items in the list is what's driving recall, we accumulate all entries per row (excluding itself) and format it into a similar matrix as we had before. If semantics is indeed overwhelmingly driving the recall, we should see that the prediction from semantic-only matrix will be similar to behavioral data matrix.

[29]:
```
# Compute semantic accumulators for each set of list matrices
accum_lsa_mat = np.zeros((lsa_mat.shape[0],lsa_mat.shape[1]))
accum_w2v_mat = np.zeros((w2v_mat.shape[0],w2v_mat.shape[1]))
```

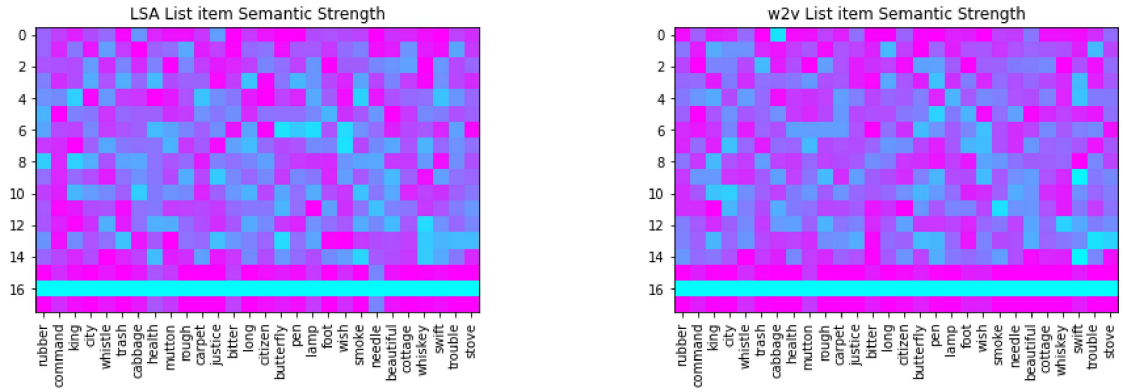
```
for i, crit in enumerate(list_order):
    lsa_sum = np.sum(list_lsa_mat[crit],axis=1) - 1
    w2v_sum = np.sum(list_w2v_mat[crit],axis=1) - 1
    accum_lsa_mat[:list_lsa_mat[crit].shape[0]-1,i] = lsa_sum[1:]
    accum_w2v_mat[:list_w2v_mat[crit].shape[0]-1,i] = w2v_sum[1:]
    accum_lsa_mat[-1,i] = accum_lsa_mat[-3,i] = lsa_sum[0]
    accum_w2v_mat[-1,i] = accum_w2v_mat[-3,i] = w2v_sum[0]

# Matrix heatmap for sum of internal semantic strength of list items
f,axes = plt.subplots(1,2,figsize=(16,4))
plt.sca(axes[0])
axes[0].title.set_text("LSA List item Semantic Strength")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(accum_lsa_mat,cmap='cool',interpolation='nearest')
plt.sca(axes[1])
axes[1].title.set_text("w2v List item Semantic Strength")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(accum_w2v_mat,cmap='cool',interpolation='nearest')
plt.show()
```



It is not obvious at this point that semantics is a good predictor for recall behavior, as we have simply summed cosine values across each row in the within-list matrices to produce each column in this new matrix, some cells have values > 1. Next we normalize each column within [0,1]

[30]:
```
# Store original
orig_accum_lsa_mat = accum_lsa_mat.copy()
orig_accum_w2v_mat = accum_w2v_mat.copy()

# Normalize each column in matrix to [0-1]
accum_lsa_mat = accum_lsa_mat / accum_lsa_mat.max(axis=0)
accum_w2v_mat = accum_w2v_mat / accum_w2v_mat.max(axis=0)
```

```
# Matrix heatmap for sum of internal semantic strength of list items
f,axes = plt.subplots(1,2,figsize=(16,4))
plt.sca(axes[0])
axes[0].title.set_text("LSA List item Semantic Strength")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(accum_lsa_mat,cmap='cool',interpolation='nearest')
plt.sca(axes[1])
axes[1].title.set_text("w2v List item Semantic Strength")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(accum_w2v_mat,cmap='cool',interpolation='nearest')
plt.show()
```

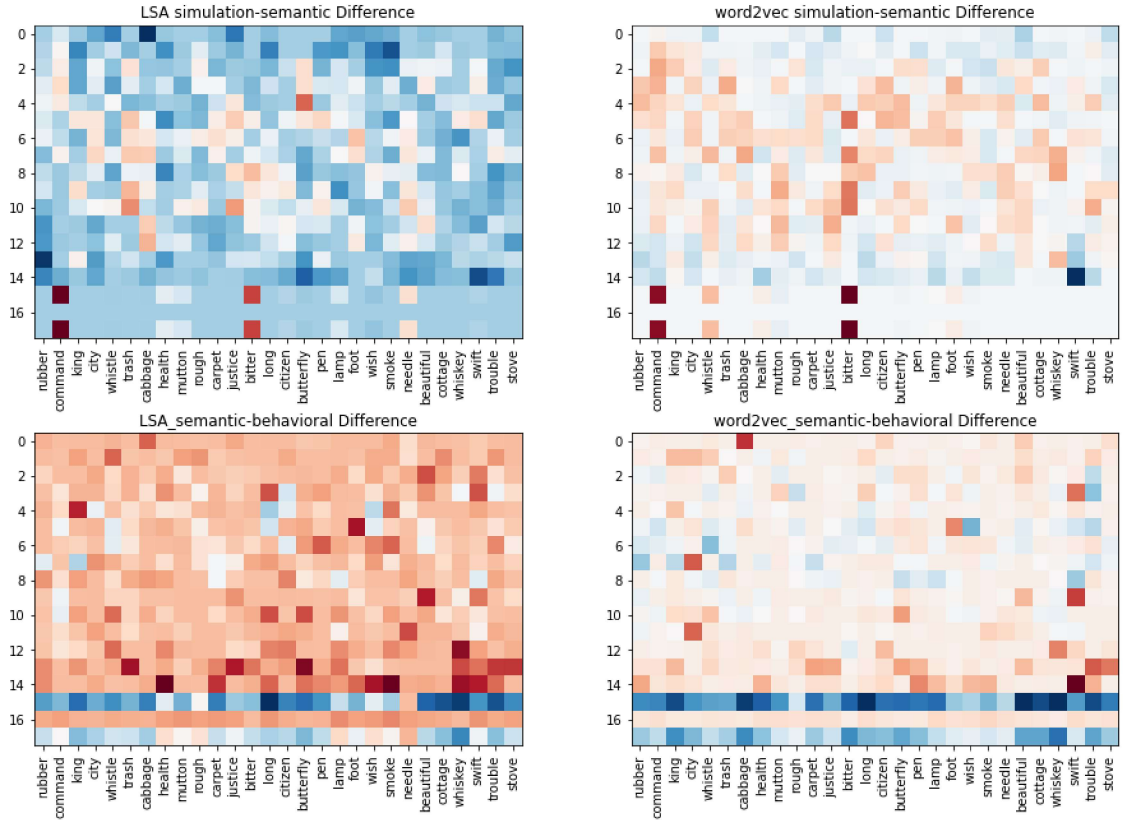


NOTE: To better model the behavioral/simulation results, there is probably some value above which we simply recall the item (instead of treating each cell value as a literal % recall)

To better illustrate the similarity between the above semantic-only predictor for recall, we run the squared-error plot as before

[31]:
```
f,axes = plt.subplots(2,2,figsize=(16,8))
f.subplots_adjust(top = 0.99, bottom=0.01, hspace=0.3, wspace=0.1)
# simulation-semantic (LSA) squared difference (preserving sign)
plt.sca(axes[0][0])
axes[0][0].title.set_text("LSA simulation-semantic Difference")
plt.xticks(ticks=range(len(cmr_list_order)),labels=list_order,rotation=90)
diff_mat = lsa_mat-accum_lsa_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# simulation-semantic (word2vec) squared difference (preserving sign)
plt.sca(axes[0][1])
axes[0][1].title.set_text("word2vec simulation-semantic Difference")
plt.xticks(ticks=range(len(cmr_w2v_list_order)),labels=list_order,rotation=90)
diff_mat = w2v_mat-accum_w2v_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
```

```
# semantic(LSA)-behavioral squared difference (preserving sign)
plt.sca(axes[1][0])
axes[1][0].title.set_text("LSA_semantic-behavioral Difference")
plt.xticks(ticks=range(len(cmr_list_order)),labels=list_order,rotation=90)
diff_mat = accum_lsa_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# semantic(word2vec)-behavioral squared difference (preserving sign)
plt.sca(axes[1][1])
axes[1][1].title.set_text("word2vec_semantic-behavioral Difference")
plt.xticks(ticks=range(len(cmr_w2v_list_order)),labels=list_order,rotation=90)
diff_mat = accum_w2v_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
plt.show()
```

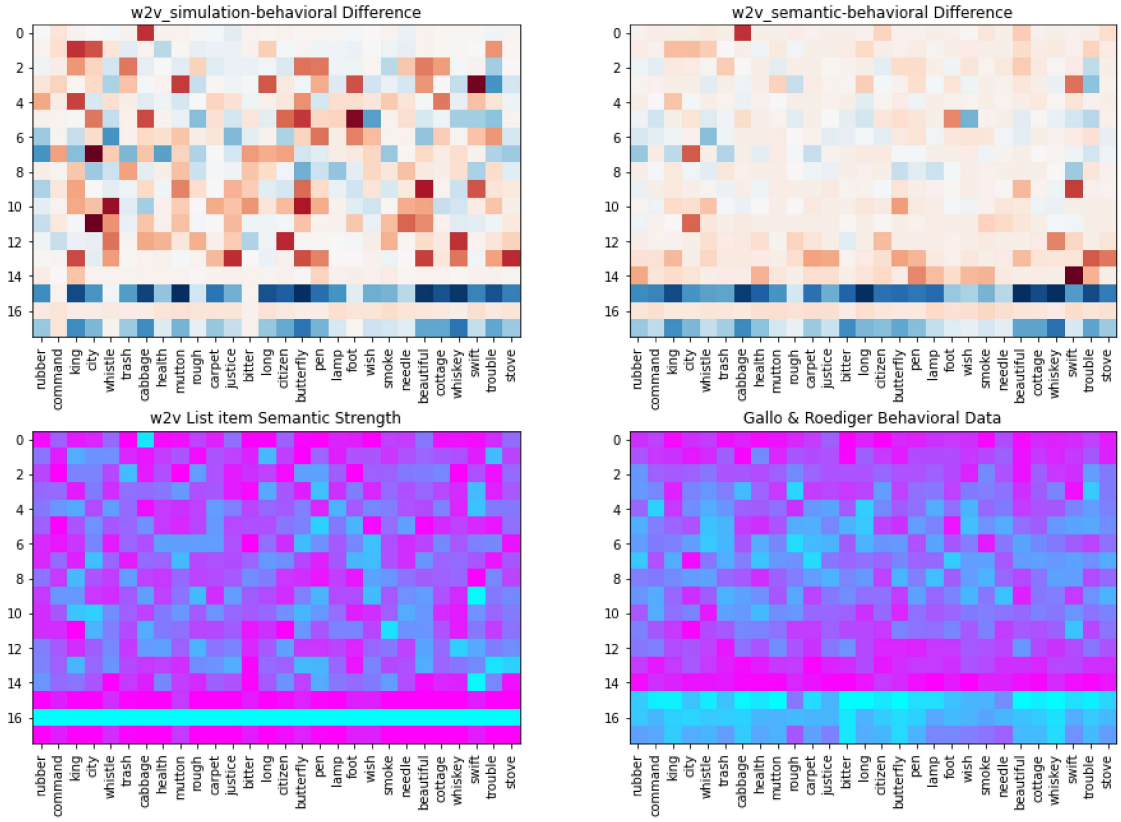


1.3.1 Semantics overwhelming

Remarkably (or maybe not), we see that the word2vec semantic matrix lies closer to behavioral results - at least for recall of items within the list but highly over-recalls the critical lure. Below is the word2vec_semantic-behavioral and word2vec_simulation-behavioral plots side-by-side to show

the difference. Furthermore, we also plot the word2vec_semantic matrix beside the behavioral data matrix for comparison.

[32]:
```
f,axes = plt.subplots(2,2,figsize=(16,8))
f.subplots_adjust(top = 0.99, bottom=0.01, hspace=0.3, wspace=0.1)
# word2vec-behavioral squared difference
plt.sca(axes[0][0])
axes[0][0].title.set_text("w2v_simulation-behavioral Difference")
plt.xticks(ticks=range(len(cmr_w2v_list_order)),labels=list_order,rotation=90)
diff_mat = w2v_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# semantic(word2vec)-behavioral squared difference (preserving sign)
plt.sca(axes[0][1])
axes[0][1].title.set_text("w2v_semantic-behavioral Difference")
plt.xticks(ticks=range(len(cmr_w2v_list_order)),labels=list_order,rotation=90)
diff_mat = accum_w2v_mat-behavioral_mat
plt.imshow(np.multiply(np.sign(diff_mat),np.
 ↪square(diff_mat)),cmap='RdBu',interpolation='nearest')
# Word2vec semantic matrix
plt.sca(axes[1][0])
axes[1][0].title.set_text("w2v List item Semantic Strength")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(accum_w2v_mat,cmap='cool',interpolation='nearest')
# Behavioral data matrix
plt.sca(axes[1][1])
axes[1][1].title.set_text("Gallo & Roediger Behavioral Data")
plt.xticks(ticks=range(len(list_order)),labels=list_order,rotation=90)
plt.imshow(behavioral_mat,cmap='cool',interpolation='nearest')
plt.show()
```



Although the top-right plot appears to show that semantics within the word2vec space is a good model for recall within list items, it does not perform as well near the edges (start & end of list) which is a region where CMR simulation does well in.

[ ]: