Citynet

duw

11/12/2019

The goal of this assignment is to study your fluency responses by visualizing them as a network and analyzing its correspondence with different natural representations.

Overview

This assignment consists of 2 steps.

1. Visualize network.
2. Compare network to natural representations.

Step I - Visualize network

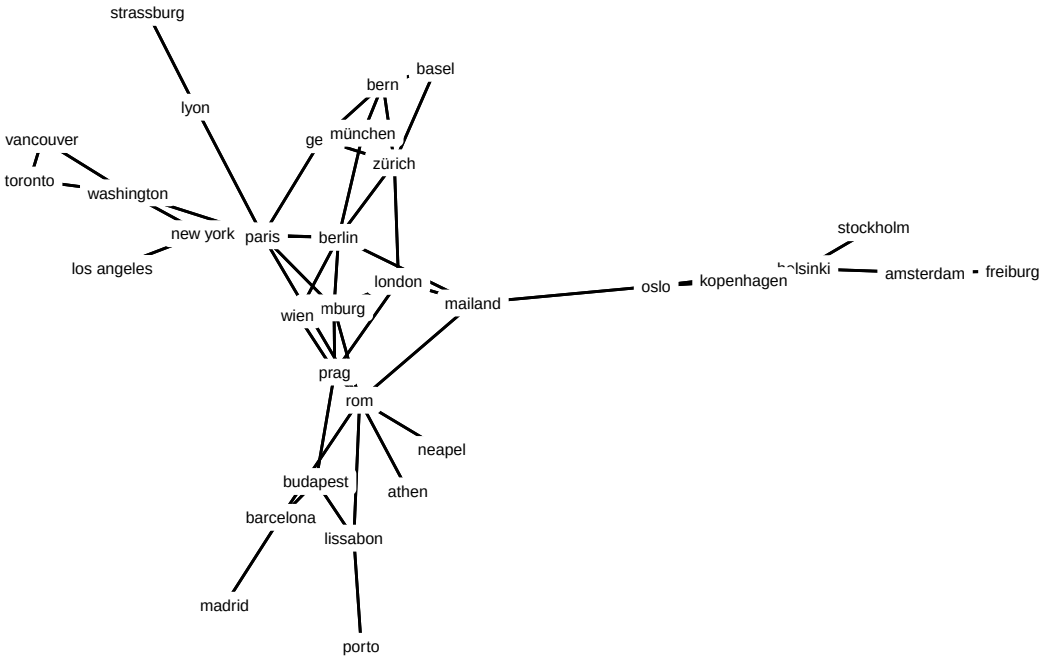
1. The first step consists of plotting the network. Download the data from the website and load it into R using the code below (you may have to adapt the path in the `download.file()`-function. The data is already processed - more on that during the seminar. You merely have to concern yourself with plotting and analyzing the data.

```
# download and load fule
download.file('https://www.dirkwulff.org/courses/2019_networks/data/city_edges.RDS', '~/Downloads/city_
edges <- readRDS('~/Downloads/city_edges.RDS')
```

2. Now plot the network using the following code. Note: You probably have to first install (`install.packages()`) and load (`library()`) the load the following packages: `igraph`, `ggraph`, `tidyverse`, and - while you're at it - `maps`.

```
# extract TRUE edges and load into igraph object
edgelist = edges %>% filter(edge) %>% select(node_i, node_j) %>% as.matrix()
graph = graph_from_edgelist(edgelist)

# plot graph
ggraph(graph) +
  geom_edge_link() +
  geom_node_label(aes(label = V(graph)$name),
                  label.size = unit(0.2, "lines"),
                  label.padding = unit(.2, "lines"),
                  size = 2.2) +
  theme_graph(base_family = "Roboto Condensed")
```

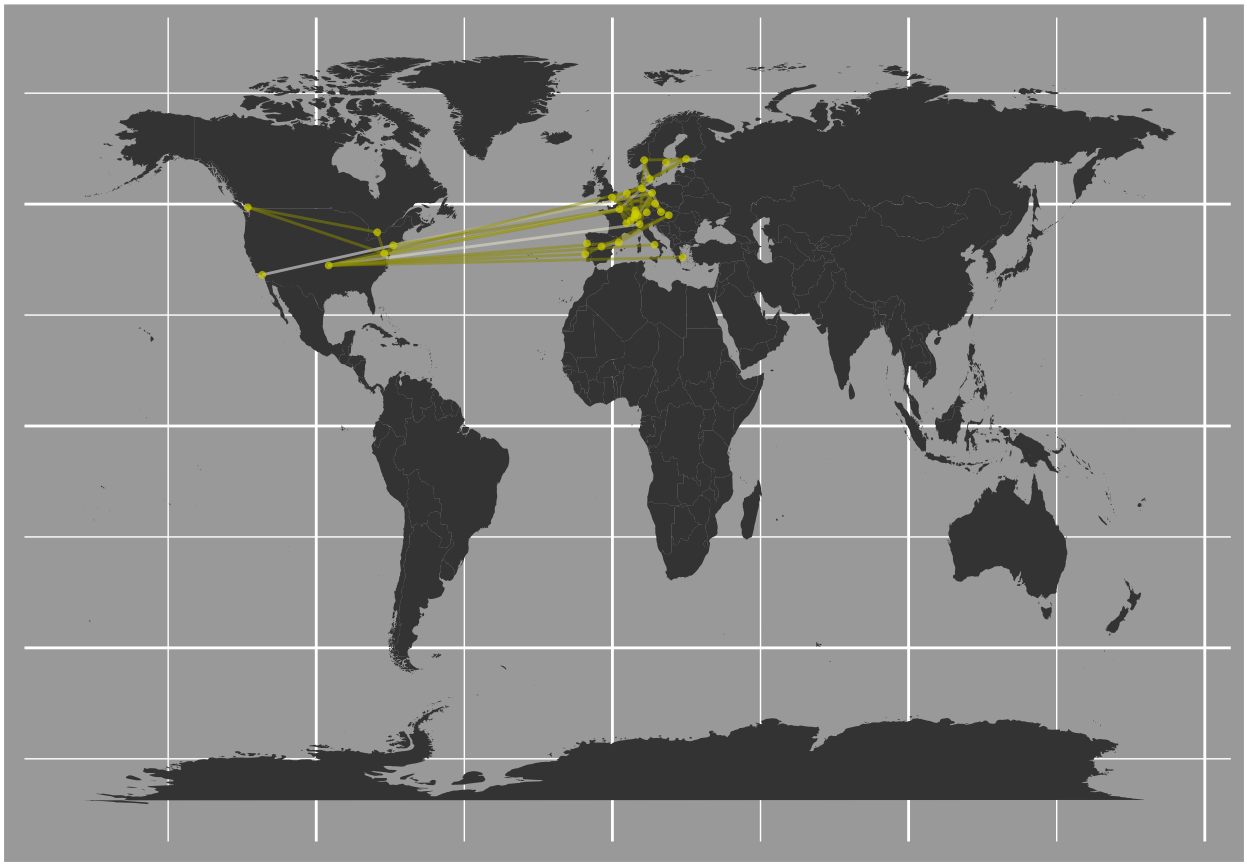


3. In the next section, you will evaluate how the network aligns with natural representations of cities such as their locations in the orthographic space, representing the similarity of city word spellings, and their location on the map. The latter is relatively straightforward to illustrate in R. Points represent your responses.

```
# load word map
map_world <- map_data("world")

# extract positions
pos_i = edges[,c('node_i','lon_i','lat_i')]
pos_j = edges[,c('node_j','lon_j','lat_j')]
names(pos_i) = names(pos_j) = c('node','lon','lat')
pos <- unique(rbind(pos_i, pos_j))

# plot map and network
ggplot() +
  geom_polygon(data = map_world, aes(x = long, y = lat, group = group)) +
  theme(panel.background = element_rect(fill = "#999999"),
        plot.background = element_rect(fill = "#999999"),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank(),
        legend.position = "none") +
  geom_segment(data = edges %>% filter(edge), aes(x = lon_i, y = lat_i, xend = lon_j, yend = lat_j, col
  scale_color_gradient(low="yellow4", high="white") +
  geom_point(data = pos, aes(x = lon, y = lat), color = 'yellow3', cex = .7, alpha = .7)
```



Step II - Compare network to natural representations

1. If cities were internally represent as a map, one would expect that inferred edges show shorter distances than inferred non-edges. The map illustration, however, shows that there are many short, but also several long distance edges in the network, making it difficult to decide whether edges coincide with short distances or not. Try to resolve this by calculating the average (`mean` and `median`) geographic distance (`geo_dist`) for edges and non-edges.

```
# calculate average distances of edges and non-edges
edges %>%
  group_by(edge) %>%
  summarize(mean_geo_dist = mean(geo_dist),
            median_geo_dist = median(geo_dist))
```

```
## # A tibble: 2 x 3
##   edge  mean_geo_dist median_geo_dist
##   <lgl>         <dbl>           <dbl>
## 1 FALSE          42.3           18.0
## 2 TRUE           29.5            8.97
```

2. An alternative organizing principle for cities, could be the orthographic space of distances between city word forms. This representation would assign *Bern-Berlin* a small distance as the names of these cities differ only by a few letters and *Sankt Petersburg-Moskau* a large distance as the names of these cities differ in almost all of their letters. Similar to the previous analysis, one would expect shorter distances for edges than non-edges in the orthographic space, if your internal representation was based on orthographic information. Calculate the average (`mean` and `median`) edit distance (`edit_dist`) and the average proportion of identical initials (`same_intitial`) for edges and non-edges.

```r
# calculate average distances of edges and non-edges
edges %>%
  group_by(edge) %>%
  summarize(mean_geo_dist = mean(geo_dist),
            median_geo_dist = median(geo_dist),
            mean_orth_dist = mean(edit_dist),
            median_orth_dist = median(edit_dist),
            mean_same_init = mean(same_initial))
```

```
## # A tibble: 2 x 6
##   edge  mean_geo_dist median_geo_dist mean_orth_dist median_orth_dist
##   <lgl>         <dbl>           <dbl>          <dbl>            <dbl>
## 1 FALSE          42.3           18.0            6.96                7
## 2 TRUE           29.5            8.97           6.63                7
## # ... with 1 more variable: mean_same_init <dbl>
```

3. Looks like both, geographic and orthographic distance are smaller for edges than for non-edges, suggesting that the organization of your internal representation is affected by both. This leads to two questions: First, what is the relative strength of the effect for each representation? And second, how independent are the two representations? That is, if one of the two only shows a small effect, than maybe it is driven by a positive correlation between the representations. Test both. Use the function below to calculate cohen's d for the difference between edges and non-edges for both geographic and orthographic distance. Then determine the correlation between geographic and orthographic distances using `cor(XX, YY, method = 'spearman')`. What do think? Which of the two governs your internal representation?

```r
# calculate cohens d
edges %>%
  summarize(geo_effect = cohens_d(geo_dist[!edge], geo_dist[edge]),
            orth_effect = cohens_d(edit_dist[!edge], edit_dist[edge]),
            geo_orth_cor = cor(geo_dist, edit_dist, method = 'spearman'))
```

```
## # A tibble: 1 x 3
##   geo_effect orth_effect geo_orth_cor
##        <dbl>       <dbl>        <dbl>
## 1      0.291       0.189        0.267
```