DimensionalityReductionSolutions

September 1, 2018

1 Solutions to Dimensionality Reduction

2 Dimensionality Reduction Task

- Use PCA from MultivariateStats.jl, to reduce 100 dimensional word embedding down to 3,2 and 1 dimensions.
- Plot these using Plots.jl, coloring acording to class

2.1 Tips:

- `plotly` is a good backend for 3D Plotting.
- The command `scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)`
- will plot a 3D scatter plot
- coloring each point according to the numerical array `classes`
- and putting a tooltip on each point, according to the string array `all_words`

3 First we loadup some data

For the the example presented here, we will use a subhset of Word Embedding, trained using Word2Vec.jl. These are 100 dimensional vectors, which encode syntactic and semantic information about words.

You can download the datased from here, and load it up with JLD as shown below. (or just load it directly if you have cloned the notebooks)

Example code for the loading, together with the words sorted into their original classes is below.

```
In [1]: using JLD
        countries = ["afghanistan","algeria","angola","arabia","argentina","australia","banglad
        usa_cities = ["albuquerque","atlanta","austin","baltimore","boston","charlotte","chicag
        world_capitals = ["accra","algiers","amman","ankara","antananarivo","athens","baghdad"
        animals = ["alpaca","camel","cattle","dog","dove","duck","ferret","goldfish","goose","g
        sports = ["archery","badminton","basketball","boxing","cycling","diving","equestrian",


        words_by_class = [countries, usa_cities, world_capitals, animals, sports]
```

```
all_words = vcat(words_by_class...)
classes = vcat(((1:5) .* ones.(length.(words_by_class)))...);
embeddings = load("../assets/ClusteringAndDimensionalityReduction.jld", "embeddings")
```

Out[1]:
```
Dict{String,Array{Float32,1}} with 185 entries:
  "ferret"       => Float32[0.0945707,-0.435267,0.0109875,-0.107674,0.169001,-0
  "gymnastics"   => Float32[-0.269173,-0.343412,-0.00603042,-0.186179,0.0342606
  "vegas"        => Float32[-0.00530534,-0.264874,0.0167432,-0.289836,-0.14033,
  "archery"      => Float32[0.0279714,-0.485648,0.105468,-0.0696941,0.182807,-0
  "jacksonville" => Float32[-0.418758,-0.0284594,0.00847164,-0.0989162,0.098186
  "ankara"       => Float32[-0.139109,0.0872892,0.749557,-0.0308427,-0.0936718,
  "pentathlon"   => Float32[-0.357405,-0.379595,-0.134314,-0.31008,-0.0245871,-
  "seoul"        => Float32[0.0274904,-0.153844,-0.0936614,-0.0269344,-0.091449
  "china"        => Float32[0.132423,-0.515862,-0.0381339,-0.287565,-0.285202,-
  "korea"        => Float32[0.236904,-0.128355,-0.0816942,-0.0702621,-0.148426,
  "argentina"    => Float32[-0.113967,-0.437523,-0.226014,-0.439572,-0.230062,-
  "mozambique"   => Float32[0.309411,-0.13457,-0.632055,-0.309943,0.040591,0.11
  "iraq"         => Float32[-0.260673,0.0356129,0.104878,0.103836,-0.17918,-0.3
  "baku"         => Float32[0.182572,0.156322,0.225807,-0.0933851,-0.246997,-0.
  "jakarta"      => Float32[0.13052,-0.408592,-0.0138496,-0.415052,0.21523,-0.0
  "bogotá"       => Float32[-0.26368,-0.292844,-0.338501,-0.278793,-0.0690988,0
  "sacramento"   => Float32[-0.217914,-0.116757,-0.213111,-0.13627,-0.0241341,-
  "dhaka"        => Float32[-0.0264262,-0.256298,0.0922423,-0.711511,-0.329286,
  "kyiv"         => Float32[-0.0527193,0.219892,-0.298013,-0.594799,-0.452732,-
  "houston"      => Float32[-0.301442,-0.133911,-0.17504,-0.0391225,-0.0525875,
  "italy"        => Float32[0.246153,-0.0510639,-0.143408,-0.149572,-0.229163,-
  "francisco"    => Float32[-0.342338,-0.200734,-0.347174,-0.228947,-0.125513,-
  "baghdad"      => Float32[-0.309163,0.00524779,0.287937,0.0294381,-0.173093,-
  "dog"          => Float32[0.0509182,-0.479764,0.0209584,-0.0409415,0.0650602,
  "kabul"        => Float32[-0.0282727,-0.108688,0.249284,0.119064,-0.163644,-0
                 =>
```

In [2]:
```
using MultivariateStats
using Plots
plotly()
```

Out[2]:
```
Plots.PlotlyBackend()
```

In [3]:
```
embeddings_mat = hcat(getindex.([embeddings], all_words)...)
```

Out[3]:
```
100Œ185 Array{Float32,2}:
  0.0386423   -0.0747454     -0.194131    -0.0949871   0.0184777
 -0.0707636    0.00147601    -0.521243    -0.540243   -0.0992318
  0.122178    -0.030897       0.0806444    0.0674903   0.343439
  0.187411    -0.201719      -0.237717    -0.0968779  -0.113297
 -0.215721    -0.181733       0.125805     0.277859    0.254373
 -0.33405     -0.0827407     -0.202835     0.153194    0.359169
  0.198505     0.356985      -0.194464    -0.0815657   0.332574
  0.290666     0.204581      -0.210431    -0.253662   -0.548761
```

```
 -0.264896    -0.240784        0.11638      0.295445    0.0797238
 -0.370904    -0.276216        0.0468465    0.0898132  -0.0984195
 -0.140316    -0.1886         0.180491    -0.147654    0.090978
 -0.0271654   -0.336009        0.00966041   0.116254    0.163717
 -0.245324    -0.002544       -0.381931    -0.646284   -0.321171

 -0.426754    -0.0195873      -0.581407    -0.29744    -0.684529
  0.194271    -0.265007       -0.319806    -0.430182    0.167392
  0.0785286    0.14811        0.0619313    0.179959   -0.0968675
 -0.401404    -0.247286        0.122847     0.146193   -0.17439
  0.00592929  -0.063444       -0.0992176    0.211413   -0.111647
  0.305375     0.0234759       0.00376886   0.0932082   0.0302637
 -0.176298    -0.0396247      -0.103947    -0.0945107  -0.260385
  0.0360829   -0.372389      -0.291512    -0.261196    0.148431
  0.0877882    0.0802952       0.044897     0.347259    0.079031
 -0.0831292   -0.18574        -0.127575    -0.0358119  -0.459085
  0.0365798   -0.154143       -0.393261    -0.215115   -0.0123871
  0.11823     -0.0554525       0.588549     0.334955    0.198312
```

In [4]:
```
#Direct projection -- no DR -- just throw away the information in the other axies
xs=embeddings_mat
scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)
```

3.0.1 PCA

In [5]:
```
M = fit(PCA, embeddings_mat; maxoutdim=3)
xs = transform(M, embeddings_mat)
scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)
```

In [6]:
```
M = fit(PCA, embeddings_mat; maxoutdim=2)
xs = transform(M, embeddings_mat)
scatter(xs[1,:], xs[2,:]; hover=all_words, zcolor=classes)
```

In [7]:
```
M = fit(PCA, embeddings_mat; maxoutdim=1)
xs = transform(M, embeddings_mat)
scatter(xs[1,:], ones(length(xs)); hover=all_words, zcolor=classes)
```

4 ICA

In [8]:
```
embeddings_mat_f64 = convert(Matrix{Float64}, embeddings_mat)

M = fit(ICA, Float64.(embeddings_mat_f64),5)
xs = transform(M, embeddings_mat_f64)
```

Out[8]:
```
5×185 Array{Float64,2}:
 -0.449666   0.0614239  -0.0577035     2.71363     3.16895     1.36657
  0.15085    1.48484     2.47812      -0.382752    0.0210489  -0.613272
  1.64663    1.68168    -0.381274     -0.0461585   0.0635969  -0.377507
```

```
 0.667897   0.0681079  -0.802374       -0.18597     0.436981    0.401189
 0.839049  -0.145083   -0.0118368      -0.381903   -0.159733   -0.789554
```

```
In [9]: scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)
```

5 Extension: T-SNE

- Use TSne.jl, to perform similar dimentionality reduction, and to produce plots.

T-SNE is another popluar DR method.
However, the TSne.jl package is not registered.
It is mostly maintained though. Be warned: it is sideways -- it is row major, so tanspose the inputs and outputs

You may have to play with the perplexity to get it to work well.

If you look at the resulting plots, you may note that countries are often paired uo with their captical city.

```
In [10]: using TSne
```

```
In [12]: xs = tsne(embeddings_mat', 3, 500, 1000, 20.0)'
```

```
Computing point perplexities100%|| Time: 0:00:00

WARNING: could not attach metadata for @simd loop.
WARNING: could not attach metadata for @simd loop.



Computing t-SNE  0%|                                    |  ETA: 0:05:31Computing t-SNE  1%|
```

```
Out[12]: 3Œ185 Array{Float64,2}:
 18.5321  -3.90908  -25.9997      2.78249     18.8778   35.9635    30.5813
 34.581   16.4268    -0.117784  34.6205      15.4415   -1.16452  -25.0301
 30.7113  36.721     54.6548    40.7141     -50.6902  -49.7441   -38.8731
```

```
In [14]: scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)
```