```
from scipy import spatial
import numpy as np
import h5py
import codecs

START = '<s>'
END = '</s>'
UNKNOWN = '<unk>'

f = h5py.File('embed_export1.hdf5', 'r')
embeddings = f['embed'][...]

# Map word indexes to embeddings
idx_to_embed = {}
idx = 1
for e in embeddings:
  idx_to_embed[idx] = e
  idx = idx + 1

# Quick lookup for idx <-> word
last_idx = -1
word_to_idx = {}
idx_to_word = {}
with codecs.open('data/words.dict', 'r', encoding='latin-1') as d:
  for line in d:
    l = line.split()
    idx = int(l[0])
    word = str(l[1])
    word_to_idx[word] = idx
    idx_to_word[idx] = word
    last_idx = idx
word_to_idx[START] = last_idx + 1
word_to_idx[END] = last_idx + 2
idx_to_word[last_idx + 1] = START
idx_to_word[last_idx + 2] = END

def word_to_embed(w):
  if w in word_to_idx:
    return idx_to_embed[word_to_idx[w]]
  return None

k = 8
# Find k nearest neighbors by cosine or dot
# Cosine similarity ignores magnitude, only relies on angle difference
# Dot product accounts for both magnitude and angle (may not matter too much
# in our embeddings due to max l2 norm = 1)
```

```
tword = 'germany'
target = word_to_embed(tword)
comps = []
for i, e in enumerate(embeddings):
  # NB: need i+1
  # dot = np.dot(target, e)
  sim = 1 - spatial.distance.cosine(target, e)
  comps.append((i+1, sim))

sorted_by_dot = sorted(comps, key=lambda tup: tup[1])
sorted_by_dot.reverse()

examples = ''
for i in range(1,k): # skip closest word as it is itself
  word = idx_to_word[sorted_by_dot[i][0]]
  score = '%.3f' % sorted_by_dot[i][1]
  examples = examples + word + ' (' + score + '), '
print(tword + ': ' + examples[:-2])
```