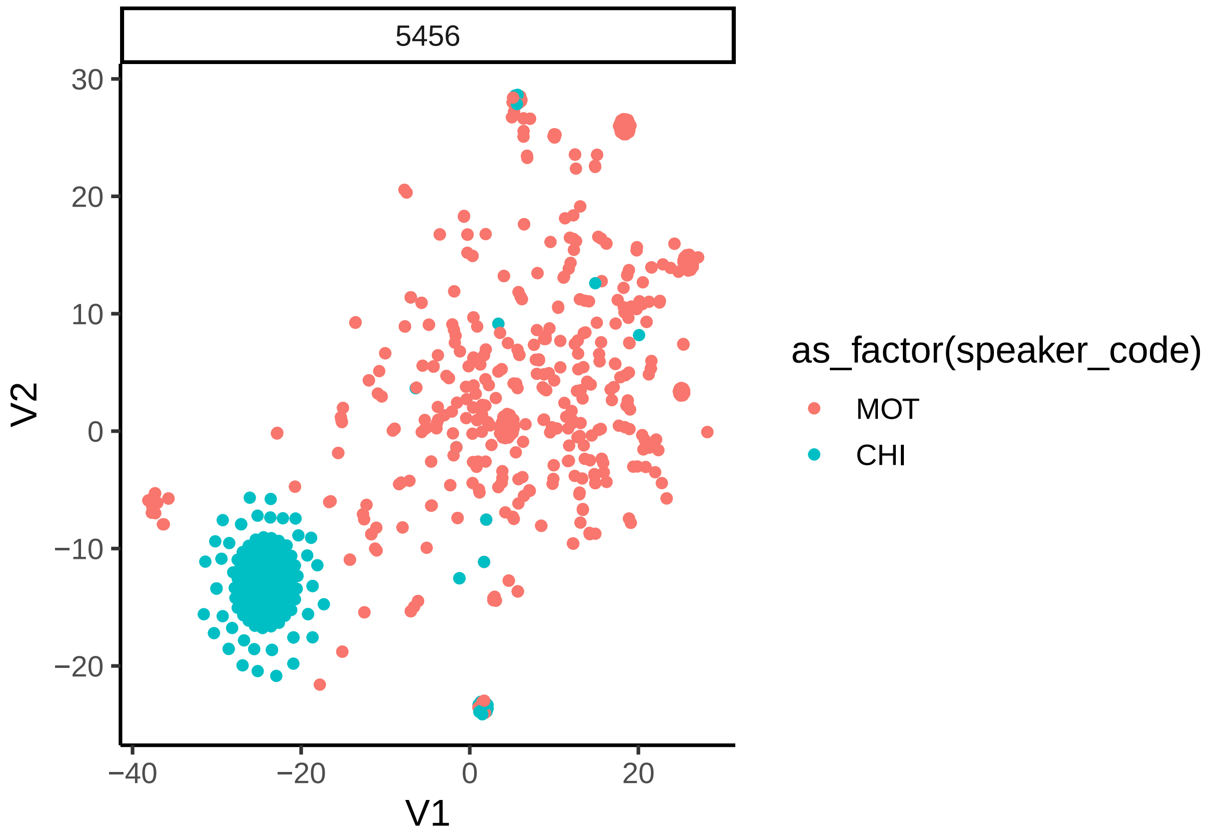
word2vec_visualizations

```r
utterances <- get_utterances(corpus = "Providence")

provcleanutts <- utterances %>%
  filter(speaker_role %in% c("Target_Child", "Mother","Father")) %>%
  arrange(transcript_id,utterance_order) %>%
  mutate(gloss = str_to_lower(gloss))

get_convo <- function(these_transcript_ids) {
  convo <- provcleanutts %>%
    filter(transcript_id %in% these_transcript_ids, gloss != "") %>%
    select(gloss, transcript_id, utterance_order, speaker_code)
  return(convo)
}


convos <- get_convo(c(5456))
```

```python
import io
import os
import gensim
from gensim import utils
import gensim.models
import gensim.models.word2vec
from gensim.test.utils import datapath
import numpy as np
import sklearn
import matplotlib
from sklearn.decomposition import IncrementalPCA
from sklearn.manifold import TSNE

model = gensim.models.Word2Vec.load("models/childes_adult_word2vec.model")


def get_vectors(convo):
  vec_length = len(model.wv['and'])
  convo_length = len(convo)
  discourse_vectors = np.zeros((convo_length, vec_length))
  for index, utt in enumerate(convo):
    sum_vector = np.zeros(vec_length)
    utt_len = len(str.split(utt))
    for word in str.split(utt):
      if word in model.wv.vocab:
        sum_vector = sum_vector + model.wv[word]
      else:
        utt_len = utt_len - 1
    if utt_len > 0:
      discourse_vectors[index] = [x/utt_len for x in sum_vector]
    else:
      discourse_vectors[index] = [0.0 for x in sum_vector]
  return discourse_vectors


def reduce_dimensions(all_convos):
  num_dimensions = 2
  all_vectors = get_vectors(all_convos)
  vectors = TSNE(n_components=num_dimensions, random_state=0).fit_transform(all_vectors)
  x_vals = [v[0] for v in vectors]
  y_vals = [v[1] for v in vectors]
  vals = np.column_stack((x_vals, y_vals, r.convos["transcript_id"], r.convos["utterance_order"]))
  return vals


convo_vecs = reduce_dimensions(r.convos["gloss"])
```

```
convos <- convos %>%
  left_join(as_tibble(py$convo_vecs), by = c("transcript_id" = "V3",
                                              "utterance_order" = "V4"))

ggplot(convos, aes(V1, V2, color = as_factor(speaker_code))) +
  geom_point() +
  facet_wrap(~ transcript_id)
```


V2
V1
as_factor(speaker_code)
MOT
CHI