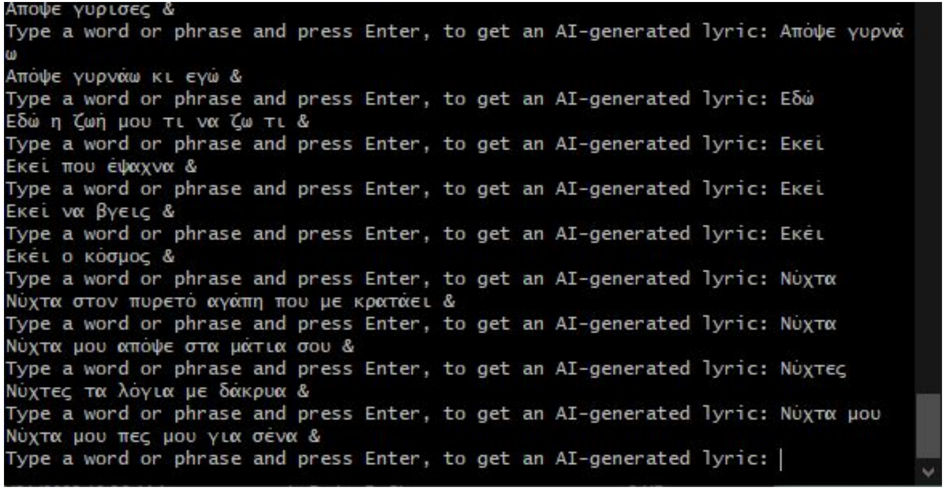
Deep Lyrics Generation

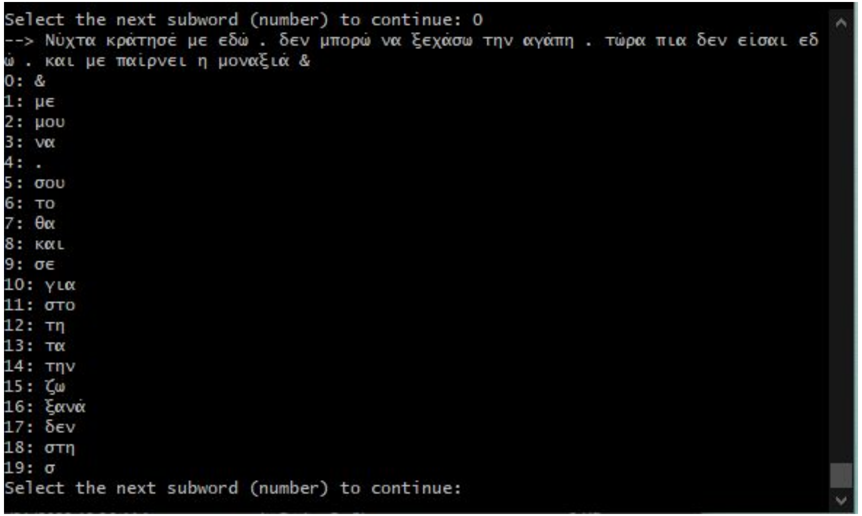
Petros Mitseas

Introduction

Using Deep Learning to generate single lyrics or verses



```
Αποψε γυρισες &
Type a word or phrase and press Enter, to get an AI-generated lyric: Απόψε γυρνά
ω
Απόψε γυρνάω κι εγώ &
Type a word or phrase and press Enter, to get an AI-generated lyric: Εδώ
Εδώ η ζωή μου τι να ζω τι &
Type a word or phrase and press Enter, to get an AI-generated lyric: Εκεί
Εκεί που έψαχνα &
Type a word or phrase and press Enter, to get an AI-generated lyric: Εκεί
Εκεί να βγεις &
Type a word or phrase and press Enter, to get an AI-generated lyric: Εκέι
Εκέι ο κόσμος &
Type a word or phrase and press Enter, to get an AI-generated lyric: Νύχτα
Νύχτα στον πυρετό αγάπη που με κρατάει &
Type a word or phrase and press Enter, to get an AI-generated lyric: Νύχτα
Νύχτα μου απόψε στα μάτια σου &
Type a word or phrase and press Enter, to get an AI-generated lyric: Νύχτες
Νύχτες τα λόγια με δάκρυα &
Type a word or phrase and press Enter, to get an AI-generated lyric: Νύχτα μου
Νύχτα μου πες μου για σένα &
Type a word or phrase and press Enter, to get an AI-generated lyric: |
```

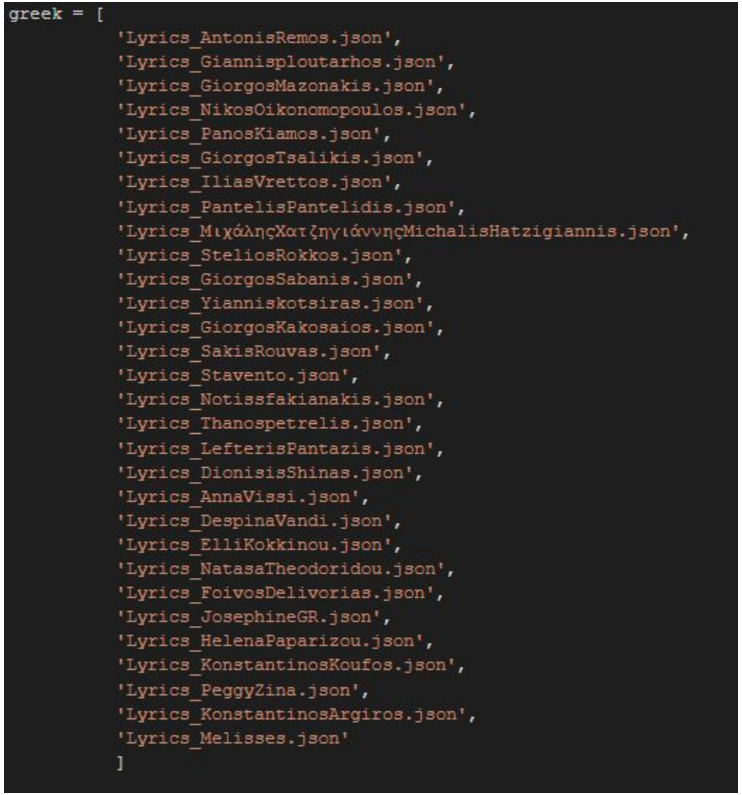


```
Select the next subword (number) to continue: 0
--> Νύχτα κράτησέ με εδώ . δεν μπορώ να ξεχάσω την αγάπη . τώρα πια δεν είσαι εδ
ώ . και με παίρνει η μοναξιά &
0: &
1: με
2: μου
3: να
4: .
5: σου
6: το
7: θα
8: και
9: σε
10: για
11: στο
12: τη
13: τα
14: την
15: ζω
16: ξανά
17: δεν
18: στη
19: σ
Select the next subword (number) to continue:
```

Dataset

Lyrics and verses from Greek Pop songs

- ~48k lyrics
- Download from Genius.com using Python API
- Basic Preprocessing (remove special characters, convert to lowercase)
- Split into single lyrics or quadruple

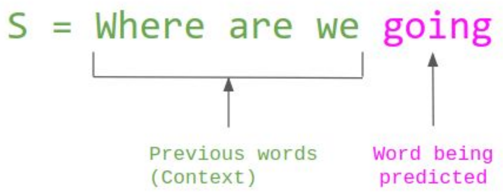


```
greek = [
          'Lyrics_AntonisRemos.json',
          'Lyrics_Giannisploutarhos.json',
          'Lyrics_GiorgosMazonakis.json',
          'Lyrics_NikosOikonomopoulos.json',
          'Lyrics_PanosKiamos.json',
          'Lyrics_GiorgosTsalikis.json',
          'Lyrics_IliasVrettos.json',
          'Lyrics_PantelisPantelidis.json',
          'Lyrics_ΜιχάληςΧατζηγιάννηςMichalisHatzigiannis.json',
          'Lyrics_SteliosRokkos.json',
          'Lyrics_GiorgosSabanis.json',
          'Lyrics_Yianniskotsiras.json',
          'Lyrics_GiorgosKakosaios.json',
          'Lyrics_SakisRouvas.json',
          'Lyrics_Stavento.json',
          'Lyrics_Notissfakianakis.json',
          'Lyrics_Thanospetrelis.json',
          'Lyrics_LefterisPantazis.json',
          'Lyrics_DionisisShinas.json',
          'Lyrics_AnnaVissi.json',
          'Lyrics_DespinaVandi.json',
          'Lyrics_ElliKokkinou.json',
          'Lyrics_NatasaTheodoridou.json',
          'Lyrics_FoivosDelivorias.json',
          'Lyrics_JosephineGR.json',
          'Lyrics_HelenaPaparizou.json',
          'Lyrics_KonstantinosKoufos.json',
          'Lyrics_PeggyZina.json',
          'Lyrics_KonstantinosArgiros.json',
          'Lyrics_Melisses.json'
          ]
```

Under the hood

How does it work? Language modeling.

Essentially, given the previous words in a sentence, try to predict the next word



$$P(S) = P(\text{Where}) \times P(\text{are} \mid \text{Where}) \times P(\text{we} \mid \text{Where are}) \times P(\text{going} \mid \text{Where are we})$$

The model can capture the underlying grammatical, syntactic and contextual information of the text and is able to generate new unseen sentences

Approaches include N-gram models, Recurrent Neural Networks and (lately) transformers such as GPT-3

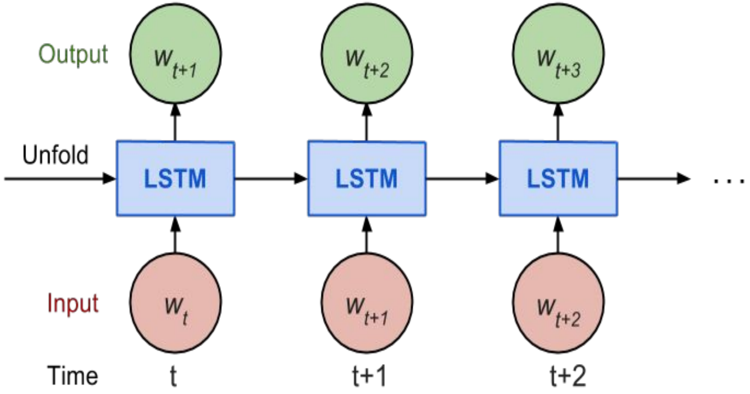
Under the hood

LSTM as a language model. The model "unfolds" as many times as the words in the lyrics.

- The output of the model is a vector with dimensions equal to the vocabulary size, where each element corresponds to the probability of this word being the next one.

- At each timestep we compare the predicted output of the model, with the true next word, and calculate the cross entropy loss.

- 1-layer with 128 neurons for single lyric modeling. 2-layer with 256 neurons for multi lyric.



Tokenization

Simple Regex Tokenization:

"Μα έλα που δεν μπορώ πλέον να αντισταθώ" -> 'Μα', 'έλα', 'που', 'δεν', 'μπορω', 'πλεον', 'να', 'αντισταθώ'

- Needs <UNK> tokens to handle unknown words
- "Suffers" from Zipf's law
- Difficult to produce rythm

Byte Pair Encoding

"Δεν έχει σίδερα η καρδιά σου να με κλείσει" -> 'δεν', 'έχει', 'σ', 'ίδ', 'ερα', 'η', 'καρδιά', 'σου', 'να', 'με', 'κλεί', 'σει'

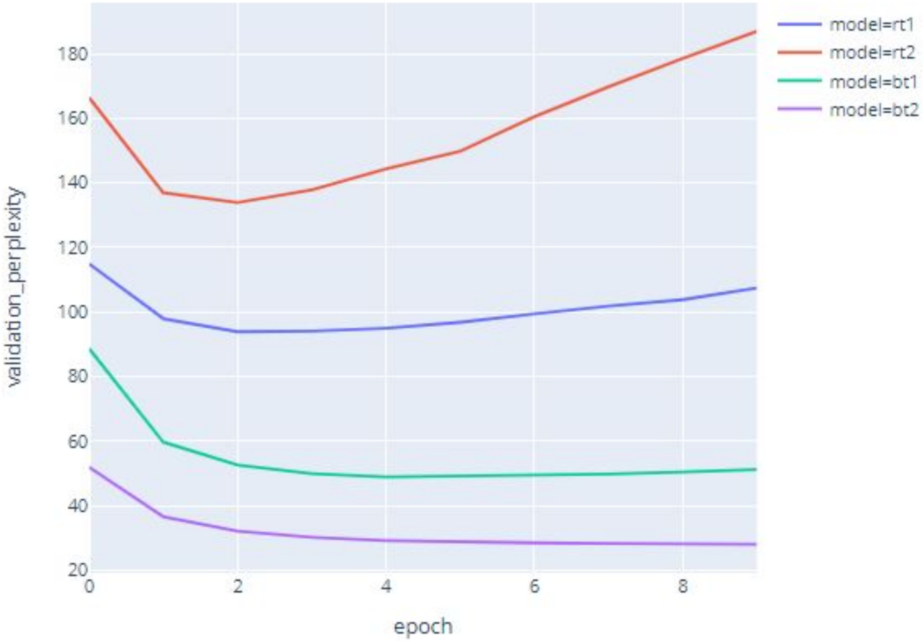
- Words are split into subwords
- Can learn suffixes and rhythm
- No need for <UNK> token
- Sometimes can join subwords into meaningless words

Evaluation

Perplexity score: Given a sentence, how likely for the model to predict that sentence.

$$PP(X) = e^{-\frac{1}{n}\sum_{i=1}^{n} \log p(x_i|x_0,x_1,\ldots,x_{i-1})}$$

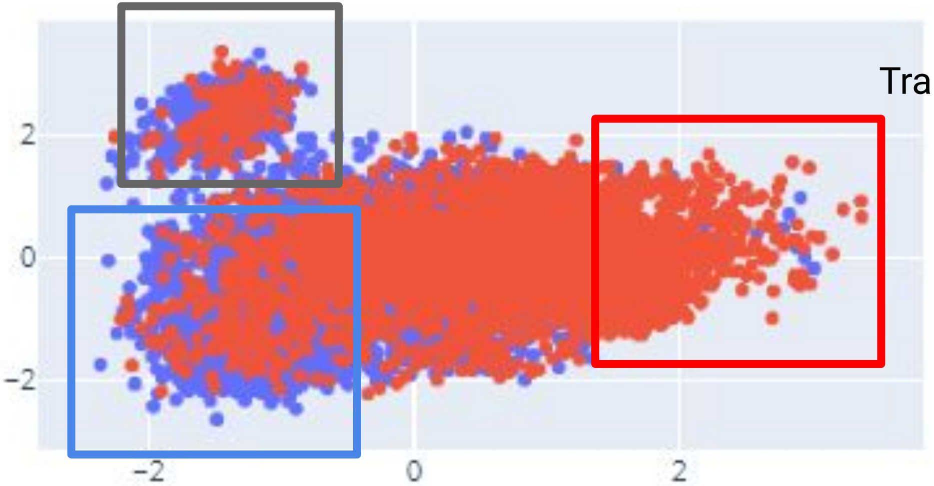
- Low perplexity usually indicates a good language model
- Closely related to cross entropy loss (easily calculated)
- Affected by the tokenization process (especially <UNK> tokens)



For validation and testing, we used lyrics from different artists than those used for training

Visualizations

PCA transformation of the LSTM's hidden layer, for whole lyric. We trained the model on both pop and trap songs.



Lyrics that end with pronoun

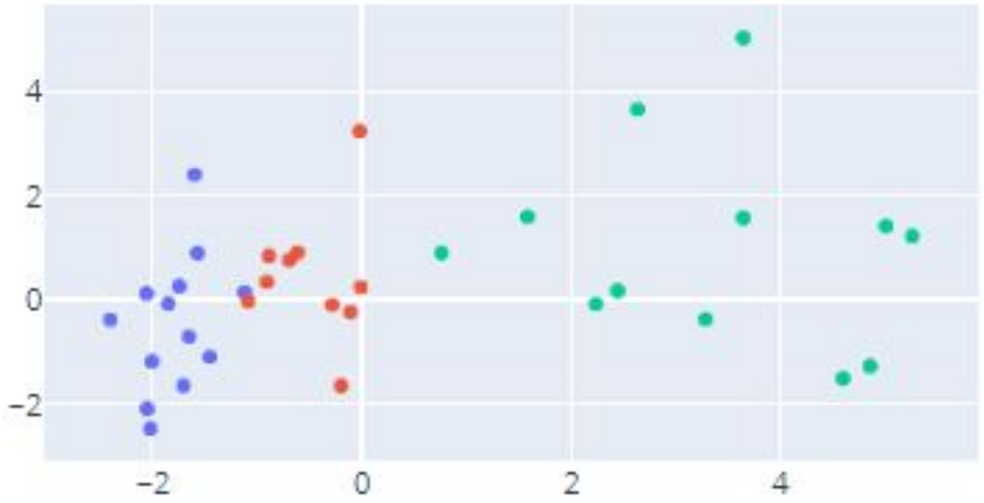
Trap lyrics

Pop lyrics

Visualizations

PCA transformation of the embedding layer for individual words.

Blue: 'πεθαίνω', 'ζω', 'αγαπώ', 'μισώ', 'λιώνω', 'ζωή', 'θάνατος', 'πιώ', 'γυρνώ', 'στιγμή', 'μωρό', 'πεθάνω', 'βλέμμα'
Red: 'νιώθω', 'δει', 'πει', 'ρθει', 'γιατί', 'θέλω', 'έλα', 'απόψε', 'σήμερα', 'αυτό'
Green: 'να', 'σε', 'με', 'μη', 'του', 'σου', 'μου', 'ο', 'η', 'το', 'μια', 'ένα'



Next steps

- Create much bigger and more curated dataset
- Buy a $$$ GPU
- Fine tune a GPT-J language model
- Share with lyric writers

Thank you