Next Word Auto-Completion

KYU CHO

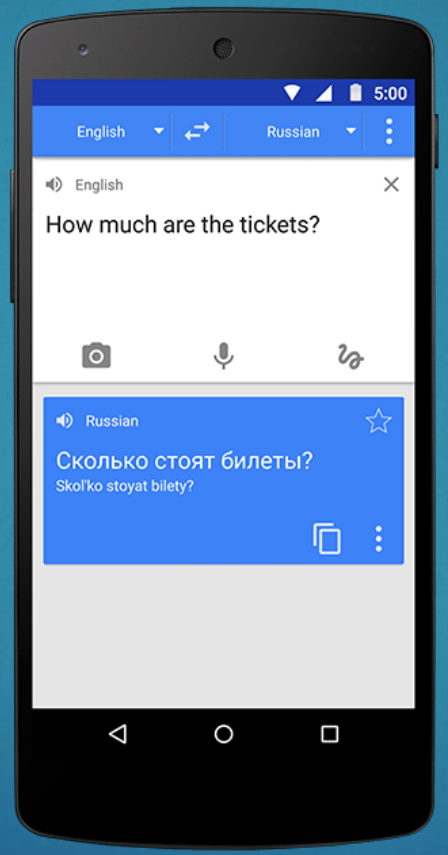
Why are language technologies needed?

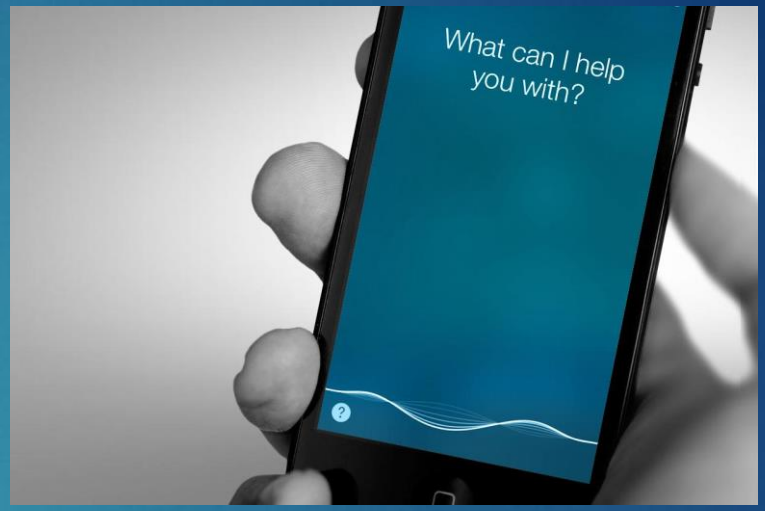
- A hallmark of human intelligence.

- Text is the largest repository of human knowledge and is growing quickly. We are living in Big Data era!!
  - emails, news articles, web pages, scientific articles, insurance claims, customer complaint letters, transcripts of phone calls, technical documents, government documents, patent portfolios, court decisions, contracts, ……
- Easy money if many companies could use computer programmes that understood text or speech in enormous amount of data quickly and accurately.

Natural Language Process (NLP) in smartphone

- Language Translation
- Sentiment Analysis
- Voice Recognition
- Question Answering
- Text Prediction
- Word Completion
- Spelling Correction
- Authorship Identification





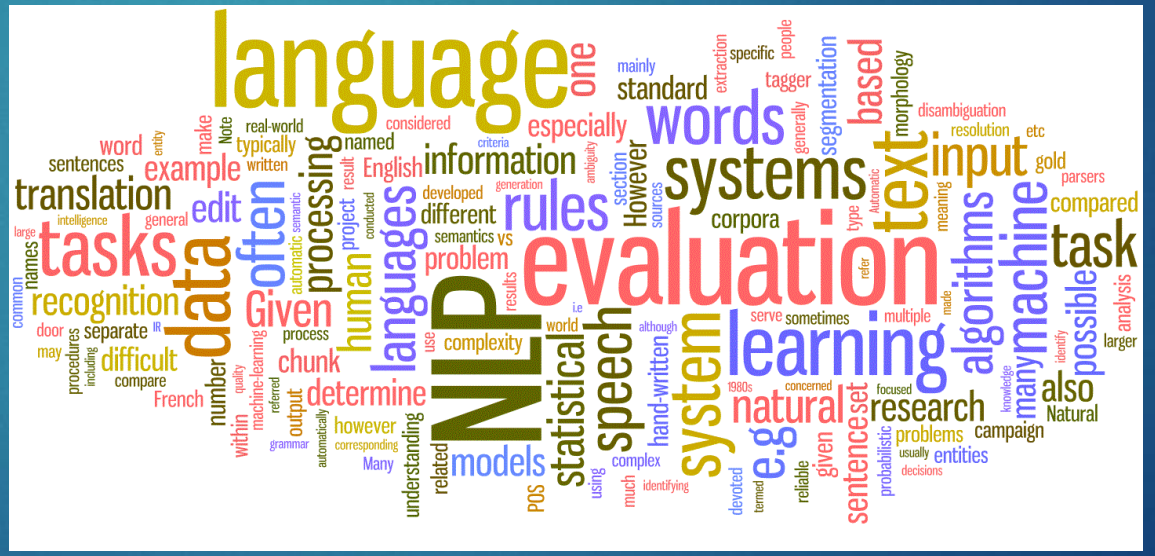
App

https://kyucho.shinyapps.io/nextword/

Human Word Prediction

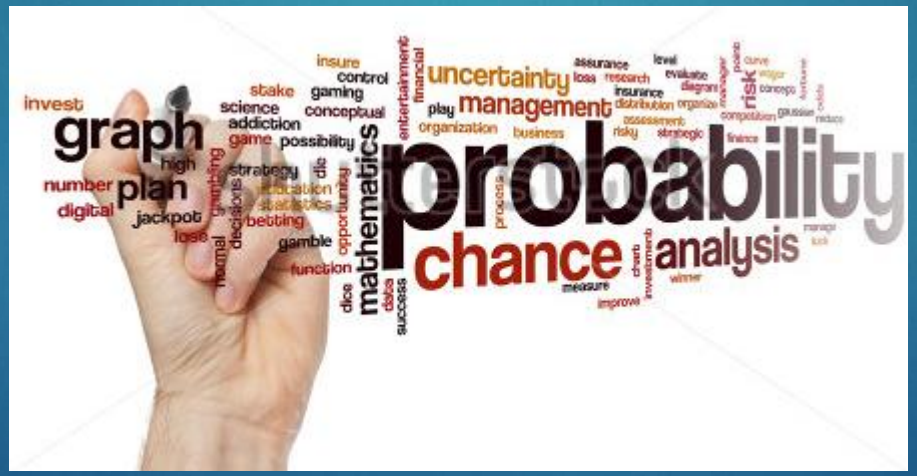
- Clearly, at least some of us have the ability to predict future words in an utterance.

- How?
  - Domain knowledge
  - Syntactic knowledge
  - Lexical knowledge



Machine Word Prediction

- A useful part of the knowledge needed to allow Word Prediction can be captured using simple statistical techniques
- The probability of a sequence of letters and words.



Computing the Probability of a Word Sequence

- Try: Computing the product of component conditional probabilities?
  - P(I want to eat Italian food) = P(I) * P(want | I) * P(to | I want)

    * P(eat | I want to) * P(Italian | I want to eat)

    * P(food | I want to eat Italian)

- Problem: The longer the sequence, the less likely we are to find it in a training data set.
  - P(Most biologists and folklore specialists believe that in fact the mythical unicorn horns derived from the narwhal)

- Solution: Maximum Likelihood Estimation(MLE) using N-grams

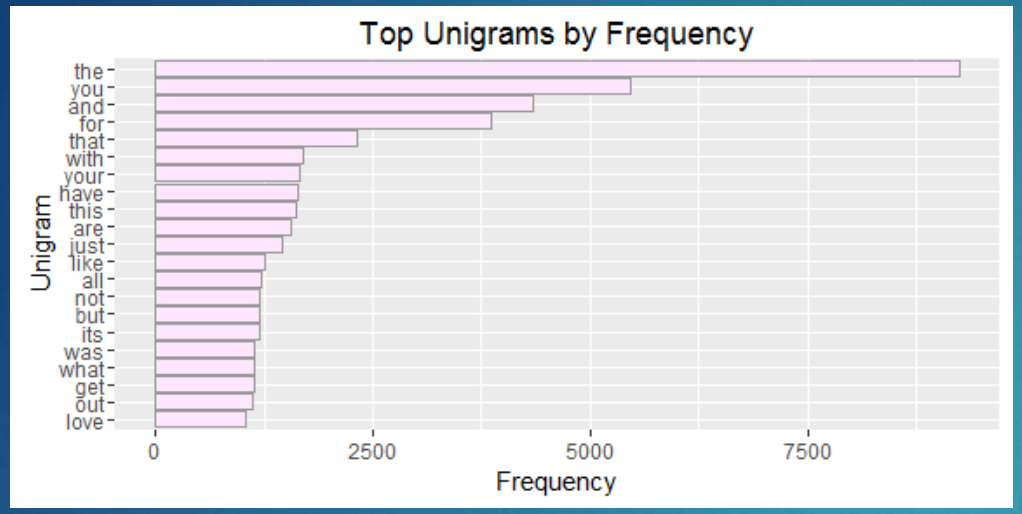
Maximum Likelihood Estimation (MLE) with N-grams

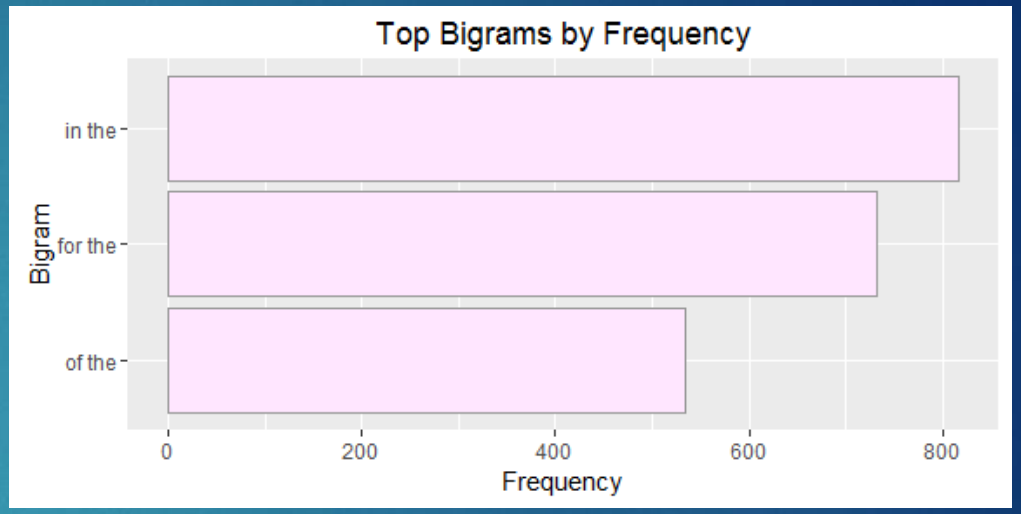
- Markov Assumption:
  - The Probability of a word depends only on the probability of a limited history.

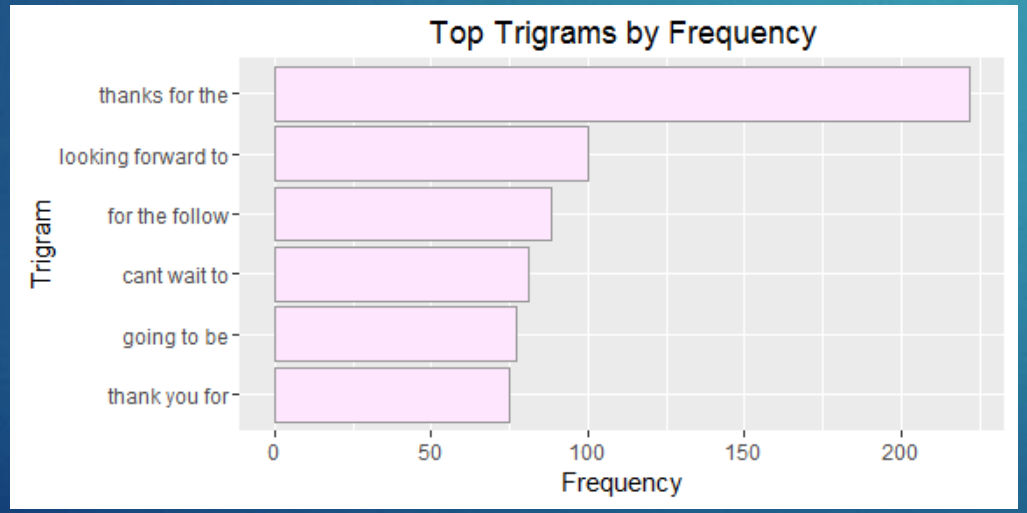
- Generalization:
  - The probability of a word depends only on the probability of the

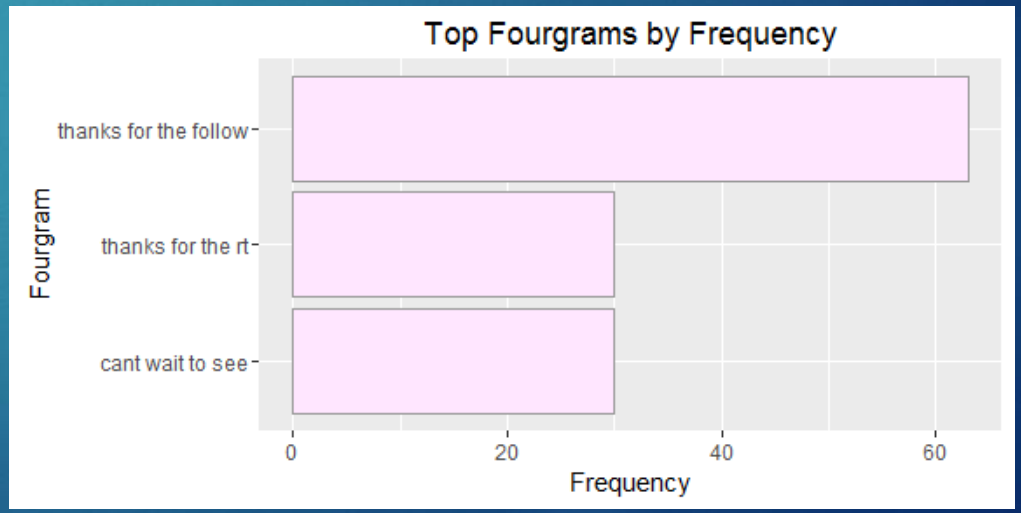
    N-previous words.
    - bi-gram, tri-grams, quad-grams,…

Plot


Top Unigrams by Frequency
Unigram
the
you
and
for
that
with
your
have
this
are
just
like
all
not
but
its
was
what
get
out
love
0
2500
5000
7500
Frequency


Top Bigrams by Frequency
Bigram
in the
for the
of the
0
200
400
600
800
Frequency


Top Trigrams by Frequency
Trigram
thanks for the
looking forward to
for the follow
cant wait to
going to be
thank you for
0
50
100
150
200
Frequency


Top Fourgrams by Frequency
Fourgram
thanks for the follow
thanks for the rt
cant wait to see
0
20
40
60
Frequency

N-grams Examples

N-gram formula

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N-1}^{n-1})$$

- Probability with Brute Force
  - P(I want to eat Italian food) = P(I) * P(want | I) * P(to | I want)
    * P(eat | I want to) * P(Italian | I want to eat)
    * P(food | I want to eat Italian)
- Probability of Bi-gram
  - P(I want to eat Italian food) = P(food | Italian)
- Probability of Tri-gram
  - P(I want to eat Italian food) = P(food | Italian) * P(food | eat Italian)
- Probability of Quad-gram
  - P(I want to eat Italian food) =
    P(food | Italian) * P(food | eat Italian) * P(food | to eat Italian)

Training and Testing

- Overly narrow corpus: probabilities don't generalize
  - Ex) brute force
- Overly general corpus: probabilities don't reflect task or domain
  - Ex) Uni-gram, bi-gram

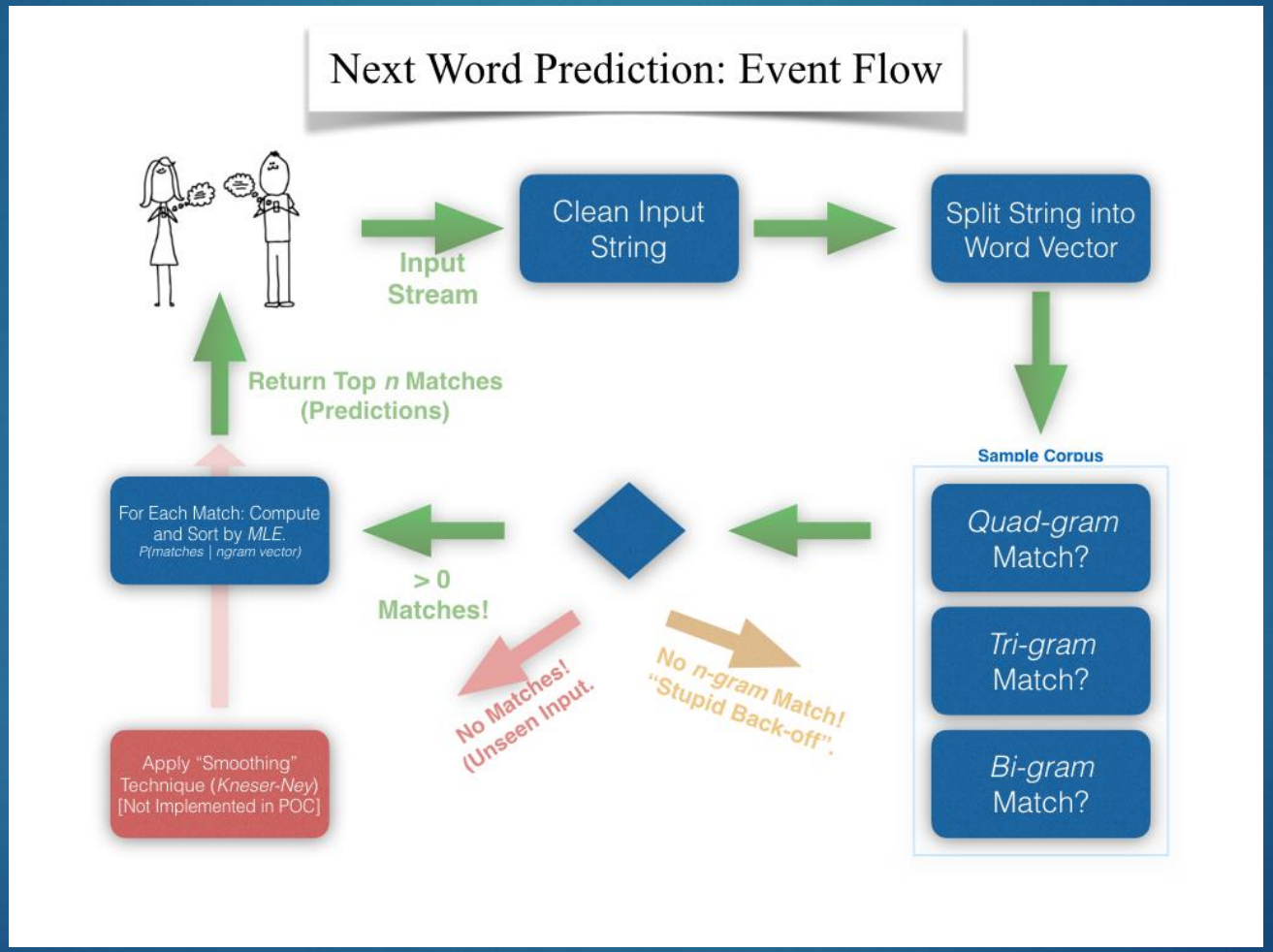
Backoff

- The higher N-gram needs more data to train.
  Thus backoff models is needed for missing value.

- **Stupid-backoff –** simple yet powerful
  If the quad-gram has no match in the data set, move to tri-gram
  If the tri-gram has no match in the data set, move to bi-gram. Etc

  "want to eat" in quad-gram data set? No then
  "to eat" in tri-gram data set? No then
  "eat" in bi-gram data set? yes then
  what's the predicted word after "eat"?

Flow Chart


Next Word Prediction: Event Flow
Input Stream
Clean Input String
Split String into Word Vector
Sample Corpus
Quad-gram Match?
Tri-gram Match?
Bi-gram Match?
No n-gram Match! "Stupid Back-off".
No Matches! (Unseen Input.
> 0 Matches!
For Each Match: Compute and Sort by MLE.
P(matches | ngram vector)
Return Top n Matches (Predictions)
Apply "Smoothing" Technique (Kneser-Ney) [Not Implemented in POC]

Table

Prediciton Table

Show 10 entries

Search:

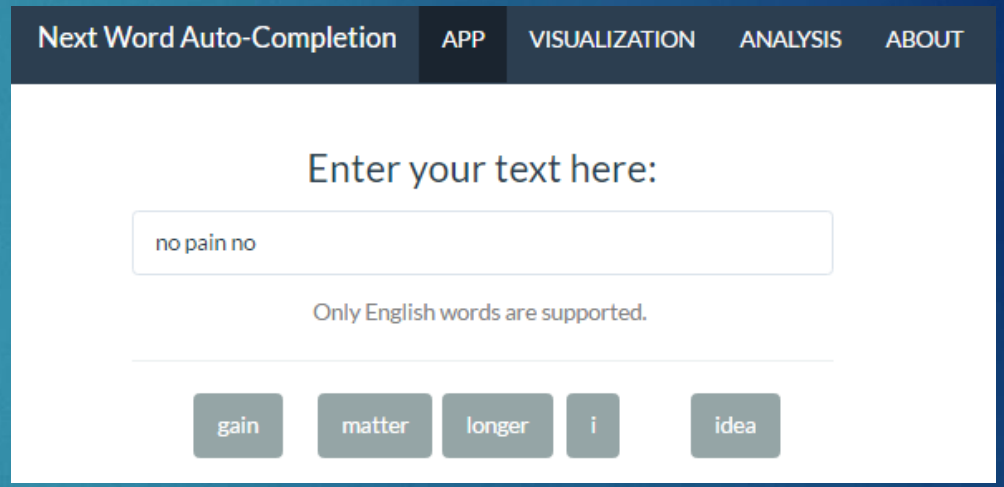
input	output	smle	counts.y
the most important	thing	0.2008	148
important	to	0.1692	1331
most important	thing	0.1568	164
the most important	part	0.0488	36
the most important	things	0.0434	32
most important	to	0.0402	42
most important	part	0.0402	42
important	than	0.0379	298
important	for	0.0371	292
most important	things	0.0354	37
input	output	smle	counts.y

Showing 1 to 10 of 80 entries

Previous 1 2 3 4 5 … 8 Next

What have I done for past few months?

- Collect data from twitter API (0.5 GB Data)
- Clean data
  - remove bad words and symbols,
  - won't -> will not, etc
- Create N-gram up to 4
- Calculate Maximum Likelihood Estimation
- Data Explanatory Analysis
- Build the Model based on
  the N-gram data set with MLE
- Develop Front-end and Back-end for UI



THANK YOU

- App
  - https://kyucho.shinyapps.io/nextword/
- Source
  - https://github.com/jamin567/DataScienceCapston
- Inquiry
  - chok20734@gmail.com