* know how text retrieval is different from database retrieval
  + Database - records (or tuples in relational databases) are typically made up of well-defined fields (or attributes)
    - bank records with account numbers, balances, names, addresses, social security numbers, dates of birth
    - Easy to compare fields with well-defined semantics to queries in order to find matches
  + Text – unstructured
    - Unstructured data is more dominant. E.g. Text in Web documents or emails, image, audio, video
    - Comparing the query text to the document text and determining what is a good match is the core issue of information retrieval
    - Exact matching of words is not enough
    - Many ways to write the same thing in a “natural language” like English
* know the basic architecture of a text retrieval system
  + **Indexing process:** build structures onto of the document collection
  + to enable searching
  + **Query process:** uses those structures and the user query to
  + produce a ranked list of documents.
  + **Text acquisition** identifies the documents that will be searched.
    - This may include crawling or scanning the Web or identifying a subset of documents in a collection.
  + **Index Creation** creates the data structures (indexes) that enable fast searching.
    - Index creation must be efficient in terms of time and space and be efficiently
    - updated when new documents are acquired. Inverted indexes are the most common form of index used for search.
  + **Text transformation** transforms documents into index terms or features.
    - Index terms can be words, phrases, names of people, dates, etc.
    - This process can involve:
    - 1. Removing stop words (common words from the document) since they contribute little to the description of the document content. Ex.” for”, “to”, “and”, etc.
    - 2. Grouping words that are derived from a common stem (called stemming). Ex. Grouping the words “fish”, “fishes”, and “fishing” to just “fish” is just one example.
    - 3. Strip unwanted characters/markup (e.g. HTML tags, punctuation, numbers, etc.).
  + **Ranking** component transforms the user’s query into terms and generates a ranked list of documents using scores based on the retrieval model.
    - Evaluation (using log data that records query execution time and user behavior) is used to evaluate the retrieval model.
* know what tokenization is, stemming, what is a stop word
  + **Tokenization** - Cut character sequence into word tokens
  + **Normalization** - Map text and query term to same form. Ex. You want U.S.A. and USA to match
  + **Stemming** - We may wish different forms of a root to match. Ex. authorize, authorization
  + **Stop words** We may omit very common words (or not). Ex. the, a, to, of
* know what an inverted index is and how to score documents quickly using index
  + An inverted index is an index data structure storing a mapping from content, such as words or numbers, to its locations in a document or a set of documents
  + See handout 1 for rankning?
* know how to measure relevance using binary relevance and vector space models
  + binary relevance
    - Precision, Mean Average Precision, Mean Reciprocal Rank
  + vector space models
    - tf, idf, tf-idf, cosign similarity
* know the major term weighting heuristics (TF, IDF, and document length normalization). Remember how TF-IDF is computed
  + TF - frequency of a term in a document
  + IDF - reflects the importance of the term in the collection of documents.
  + document length normalization –
    - One simple length normalization formula is to divide the number of occurrences by the length of the document
    - BM25
* understand probabilistic ranking principle
  + If a reference retrieval system’s response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request
  + where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose,
  + the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data.”
  + Problems:
    - doesn’t tell us how to calculate or estimate the probability of relevance
    - There is no accurate estimate for the first run probabilities
    - Index terms are not weighted
    - Terms are assumed mutually independent
* know Bayes rule and chain rule in probability
  + Bayes
  + Chain rule
    - P(A ∩ B) = P(A | B) \* P(B)
* know what a statistical language model is, what is a unigram/bigram language model
  + statistical language model - assign a probability distribution over sequences of words.
    - How likely is a given string (observation) in a given “language”
  + Unigram
    - Words are sampled independently of each other. probability of each word only depends on that word's own probability in the document
    - Best for information retrieval because of word independence
    - The probabilities in the language model predict what the next word in the sequence will be
    - the previous words have no impact on the prediction
  + Bigram
    - probabilities depend on previous words
* know why smoothing is necessary when estimating a language model and know the general idea of smoothing
  + The general approach to smoothing is to lower (or discount) the probability estimates for words that are seen in the document text and assign that “leftover” probability to the estimates for the words that are not seen in the text.
  + Smoothing is important for many reasons.
    - Assigning zero probability to possible events is incorrect.
    - Maximum likelihood estimates from your data don’t generalize perfectly to new data, so a Bayesian update from some kind of prior works better.
* be familiar with formulas for Add-1 smoothing, Dirichlet prior smoothing, and linear interpolation smoothing and their similarities and differences
  + Add-1 smoothing
    - Add 1 to every count, normalize
    - Problematic for large vocabularies
  + Dirichlet prior smoothing
    - Make smoothing depend of sample size
    - Best out-of-the-box choice for short queries
  + linear interpolation smoothing
    - instead of a fixed epsilon for all terms, let’s use probability of terms in the collection
* know how to compute the basic retrieval evaluation measures (e.g., precision, recall, and mean average precision, MRR, F1). You need to remember the formulas
  + see notes
* know Zipf’s and Heap’s law
  + Zipf’s
    - see midterm review 15/16, frequency and rarity of words
  + Heap
    - rate at which new words will appear
    - v = k x n^b
      * v = number of new words
      * k is between 10 and 100
      * n = total words
      * b usually equal to .5
* know evaluation metrics and why one metric is better than another.
  + If you recall everything, then you generate results that are not accurate, hence lowering precision.
  + If precision is high, obviously recall will be low.
  + F-measure summarizes effectiveness in a single number
  + Arithmetic mean - Will be affected more by values that are unusually large (outliers).
  + Harmonic mean harmonic mean emphasizes the importance of small values
  + Mean Average Precision (MAP)
    - Most frequently used measure in research papers
    - Average precision values at ranks of relevant docs
    - Assumes user wants to find many relevant docs
    - Biased toward top of the ranking (rank1=2\*rank2)
  + Mean Reciprocal Rank (MRR) is the average of the reciprocal ranks over a set of queries
    - very sensitive to rank position