EB5004 PLP – NEW MEDIA AND SENTIMENT MINING

ENTITY & ASPECT MINING

Dr. Fan Zhenzhen
Institute of Systems Science
National University of Singapore

Module Objectives

At the end of this module, you will be able to

- To extract key entities and aspects from opinions for closer analysis of sentiments and their targets
- To learn the main methods of entity/aspect extraction

Agenda

- The opinion target - entity and its aspects
- Tasks in ABSA
- Entity extraction
- Aspect extraction
- Recent advances

The Opinion Target and its Aspects

- An *opinion* consists of a *sentiment* (positive or negative) and a *target* (of opinion).
- Detect subjective opinion and determine its polarity is often the first step in sentiment mining.
- A common follow-up question is : what exactly do people like or not like?
- Getting the right answer requires **finer-grained** analysis, identifying opinion *target*, which can be an *entity*, or its *aspects*.
- Crucial for applications like product review analysis.

Definition of Opinion

- Recall that opinion is formally defined as a quadruple (g, s, h, t), where
  - g is the sentiment target,
  - s is the sentiment of the opinion about the target g,
  - h is the opinion holder (the person or organization who holds the opinion),
  - and t is the time when the opinion is expressed..
- In the case of entity and aspect mining, there is an additional dimension – aspect/ feature ***a***.
- Extricate quintuple (e, ***a***, s, h, t) with entity e and aspect ***a*** together representing the opinion target.

Opinion Target Extraction

- Examples:
  - *Although the **service** is not that great, I still love this **restaurant**.*
  - *The **iPhone**’s **call quality** is good, but its **battery life** is short.*
- **Entities** - names of products, services, individuals, events, and organizations
- **Aspects** – the attributes and components of entities
- Identifying them from text data is called *opinion target extraction*.
- Information Extraction (IE) tasks.

Typical tasks for aspect

- Identify subjective/opinionated sentences
- Find and extract entities and aspects that have been commented on by an opinion holder
- Determine whether the opinions are positive, negative or neutral
- Group entity/aspect synonyms
- Produce an aspect-based opinion summary of multiple reviews

ABSA subtasks

s_1 Positive s_2 Positive

S: The drinks are always *well made* and wine selection is *fairly priced*.

a_1 o_1 a_2 o_2

Subtask	Input	Output	Task Type
Aspect Term Extraction(*AE*)	S	a_1, a_2	Extraction
Opinion Term Extraction(*OE*)	S	o_1, o_2	Extraction
Aspect-level Sentiment Classification(*ALSC*)	S + a_1	s_1	Classification
	S + a_2	s_2	
Aspect-oriented Opinion Extraction(*AOE*)	S + a_1	o_1	Extraction
	S + a_2	o_2	
Aspect Term Extraction and Sentiment Classification(*AESC*)	S	(a_1, s_1), (a_2, s_2)	Extraction & Classification
Pair Extraction(*Pair*)	S	(a_1, o_1), (a_2, o_2)	Extraction
Triplet Extraction(*Triplet*)	S	(a_1, o_1, s_1), (a_2, o_2, s_2)	Extraction & Classification

How?

- The methods and features used for recognizing **entities** and **aspects** are usually different due to their individual specific characteristics.
- Main idea:
  - An opinion always has a target.
  - The target is an aspect or an entity.
  - It’s usually a noun or noun phrase (but not always).
  - => often leverage on syntactic structures to discover opinion and target relationships

ENTITY EXTRACTION

Entity Extraction

- Similar to the classic problem of Named Entity Recognition (NER) in NLP.
- Main approaches
  - Rule-based
  - Supervised statistical machine learning (e.g., HMMs, CRFs, etc.)
  - Semi-supervised approaches that only require some unambiguous seed entity names to find all entities of the same type (e.g., PU learning, Bayesian Sets, etc)
- Mature tools are available to perform NER (GATE, NLTK, Stanford NER, etc.), but not for entity extraction.
- For sentiment analysis, an additional step is usually needed to group named entities into synonyms, *entity resolution* (ER), as the entity can be mentioned in various ways.

Differences from traditional NER

- In traditional NER, the objective is to recognize all named entities of certain types in a corpus, e.g., names of people, names of organizations, etc.
- In sentiment mining, the interest is often to find the mentions of a set of desired entities, e.g. the company's own products, its competitors' products, etc.
- Solved in two steps:
  - Identify all entity mentions or entity expressions of the interested set of entities.
  - For each entity mention/expression, determine to which entity it belongs, aka *entity linking* or *entity disambiguation*, which is a special case of ER

Entity Resolution

- Two name ambiguity problems:
  - Polysemy – *Apple* for *Apple Inc*. (the maker of iPhone and iPad), or *Apple Daily* (a Hong Kong newspaper)?
  - Synonymy – *National University of Singapore* vs. *NUS*, *Volkswagen* vs. *Vwagen*, *Singapore* vs. *the little red dot*, *sound/voice/sound quality* etc.
- When the set of target entities is available, the task is to cluster entity mentions, and map them to target entities.

Entities in different types of text data

- Entity-focused corpora – online reviews of products and services.
  - Entity information can be obtained from meta-data
  - Mentions of other entities might need to be identified for comparison opinion mining
- Domain-focused corpora – forum discussions.
  - Normally focusing on discussions of a specific type of products or topics.
  - Need to perform entity extraction and linking
- Open domain corpora – like Twitter.
  - Can contain documents of **any** entity or topic, with little or no meta-data (like hashtags)
  - Obviously entity extraction and linking is needed. Most challenging.

Search helps

- In a very large corpus, keyword search is often applied first to retrieve relevant posts, using name variations of the desired entities.
- Thus, it’s better to have a clear mining objective, from which you’ll derive a comprehensive list of name variations to search the large corpus, like Twitter.
- Due to polysemy, a filtering step may be required to get posts that truly contain the desired entities.

How: Semi-Supervised Extraction

- Supervised entity extraction using HMM or CRF is still applicable here, but it requires labeling of training data, which may not always be possible.
- Therefore the semi-supervised approaches are often applied; e.g. PU learning (learning from positive and unlabeled examples), Bayesian sets
- Given a set of seed entity names, identify all entities of the same type as the seeds from a given corpus
- General idea: if they appear in similar context, they are likely to be the same kind!

PU Learning

- For example, to identify entities of phones
  - Given a set of positive examples (of seed entities), and a corpus
  - Find candidate entities from the corpus
    - Using sequences of specific POS tags: NNP (proper noun), NNPS (plural proper noun), and CD (cardinal number)
    - E.g. "Samsung/NNP Galaxy/NNP S5/CD" as a candidate
  - For each seed entity, create a TF vector for each mention of it representing a positive example, using the surrounding words context of the seed mention.
  - Do the same for each candidate entity
  - Use a PU learning algorithm like S-EM (Liu et al., 2002) to learn a classification model to label the candidates.

Li, X., L. Zhang, B. Liu, and S. Ng. Distributional similarity vs. PU learning for entity set expansion. In *Proceedings of Annual Meeting of the As-sociation for Computational Linguistics* (*ACL-2010*), 2010b.

How: Supervised Entity Linking

- To identify if a mention refers to some known entity, or none.
  - But instead of PER(person), ORG(organization), GPE(geopolitical entity), UKN(unknown), we are typically looking for products, services, and brands.
- What's required: the set of desired entities, each with its disambiguating text and type (KB); and entity expressions to be linked
- Example solution, supervised:
  - Candidate generation: for each entity expression, generate possible entities from KB using heuristic rules -> multiple (expression, entity) pairs
  - Candidate ranking: learning to rank the pairs (features such as similarity of entity name strings, similarity of context, entity type, etc.)
  - Or classification: learn a classifier to predict if the (expression, entity) pair is positive/negative
  - Case of NIL (no applicable entity from KB): when no pair is predicted to be positive.
- Main challenges: feature engineering, and labeled examples

Another complexity

- In sentiment analysis applications involving consumer products, products typically have brands and models, which can form a hierarchical relationship.
- Brand – product – model
  - e.g. Apple – iPhone – iPhone X
- Usually by separating brands and models

Opinion Holder and Time Extraction

- NER task!
- For social media data
  - the opinion holder is usually the author of the review, blog, post, etc.
  - It's trivial when the ID and date/time of the post are usually known.
- For other data, like news articles, they may need to be extracted out of text.
  - E.g. consider person and organization entities, and score them using Maximum Entropy model.

ASPECT EXTRACTION

Main approaches

1. By finding frequent nouns and noun phrases.
2. By exploiting syntactic relations
   - Syntactic dependencies depicting opinion and target relations
   - Lexico-syntactic patterns encoding entity and part/attribute relations
3. Using supervised learning

1. Frequency-based approach

- Assumption: a reasonable number of reviews about the same product or at least about the same type of products.
  - Find nouns and noun phrases using a POS tagger
  - Count their occurrence frequencies
  - Keep only the frequent ones above a experimentally determined threshold
- **It works**.
  - Aspects are usually expressed as nouns and noun phrases.
  - Vocabulary converges when people comment on the same (type of) product.
  - Irrelevant contents tend to be infrequent.

Frequency-based aspect extraction

- Frequency is the key here.
- Simple and effective. The candidate aspects are almost always the most important aspects of the product.
- Applicable to entity extraction too.
- Caution: Won’t work if the corpus has a mixture of very different products and/or if each product has only one or two reviews.

Better precision with co-occurrence

- To improve its precision, use heuristics to filter for noun phrases that are more likely to be aspects of entities.
  - E.g. for camera reviews, phrases indicating *part-of* relations, like "*of camera*", "*camera has*", "*camera comes with*", can be used to find camera components by web search
  - The discovered phrases that often co-occur with such *part-of* (*meronymy*) relation indicators are likely to be correct aspects.
  - A simplified version of Point-wise Mutual Information (PMI) can be used to compute the co-occurrence strength

$$PMI(a, d) = \frac{hits(a \wedge d)}{hits(a)hits(d)}$$

Parts or Attributes

- To distinguish components/parts from attributes
- Can use Morphological cues (e.g., "-*iness*," "*-ity*" suffixes)
- WordNet is a great resource
  - Synonyms ("*happy*" and "*joyful*"), antonyms ("*happy*" and "*sad*")
  - Hyponyms and hypernyms (*is-a* hierarchy), meronyms and holonyms (*part-whole* relation)
  - For example, we can query WordNet with the word "*camera*" and get its senses

Noun

- **S: (n) camera, photographic camera (equipment for taking photographs (usually consisting of a lightproof box with a lens at one end and light-sensitive film at the other))**
  - *direct hyponym* / *full hyponym*
  - *part meronym*
  - *direct hypernym* / *inherited hypernym* / *sister term*
- S: (n) television camera, tv camera, **camera** (television equipment consisting of a lens system that focuses an image on a photosensitive mosaic that is scanned by an electron beam)

*https://**wordnet**.princeton.edu/*

is-a

- Hyponyms (*x* is-a camera)

  - direct hyponym / **full hyponym**
    - S: (n) box camera, box Kodak (a simple camera shaped like a rectangular box)
    - S: (n) candid camera (a miniature camera with a fast lens)
    - S: (n) digital camera (a camera that encodes an image digitally and store it for later reproduction)
      - S: (n) webcam (a digital camera designed to take digital photographs and transmit them over the internet)
    - S: (n) flash camera (a camera with a photoflash attachment)
    - S: (n) motion-picture camera, movie camera, cine-camera (a camera that takes a sequence of photographs that can give the illusion of motion when viewed in rapid succession)
      - S: (n) sound camera (a movie camera that records sounds in synchrony with the visual images)
    - S: (n) point-and-shoot camera (a lightweight photographic camera with an autofocus)
    - S: (n) Polaroid camera, Polaroid Land camera (a camera that develops and produces a positive print within seconds)
    - S: (n) portrait camera (a camera with a portrait lens)
    - S: (n) reflex camera (camera that allows the photographer to view and focus the exact scene being photographed)

is-a

- Hypernyms (camera is-a *x*)

- *direct hypernym* / ***inherited hypernym*** / *sister term*
  - S: (n) photographic equipment (equipment used by a photographer)
    - S: (n) equipment (an instrumentality needed for an undertaking or to perform a service)
      - S: (n) instrumentality, instrumentation (an artifact (or system of artifacts) that is instrumental in accomplishing some end)
        - S: (n) artifact, artefact (a man-made object taken as a whole)
          - S: (n) whole, unit (an assemblage of parts that is regarded as a single entity) *"how big is that part compared to the whole?"; "the team is a unit"*
            - S: (n) object, physical object (a tangible and visible entity; an entity that can cast a shadow) *"it was full of rackets, balls and other objects"*
              - S: (n) physical entity (an entity that has physical existence)
                - S: (n) entity (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

- *direct hypernym* / *inherited hypernym* / ***sister term***
  - S: (n) photographic equipment (equipment used by a photographer)
    - S: (n) **camera**, photographic camera (equipment for taking photographs (usually consisting of a lightproof box with a lens at one end and light-sensitive film at the other))
    - S: (n) clapperboard (photographic equipment used to synchronize sound and motion picture; boards held in front of a movie camera are banged together)
    - S: (n) developer (photographic equipment consisting of a chemical solution for developing film)
    - S: (n) enlarger (photographic equipment consisting of an optical projector used to enlarge a photograph)
    - S: (n) flash, photoflash, flash lamp, flashgun, flashbulb, flash bulb (a lamp for providing momentary light to take a photograph)
    - S: (n) light meter, exposure meter, photometer (photographic equipment that measures the intensity of light)
    - S: (n) photographic paper, photographic material (light-sensitive paper on which photograph can be printed)
    - S: (n) sensitometer (a measuring instrument for measuring the light sensitivity of film over a range of exposures)

part-of

- Meronyms (*x* is part-of camera)

  - *part meronym*
    - S: (n) aperture (a device that controls amount of light admitted)
    - S: (n) camera lens, optical lens (a lens that focuses the image in a camera)
    - S: (n) delayed action (a mechanism that automatically delays the release of a camera shutter for a fixed period of time so that the photographer can appear in the picture)
    - S: (n) diaphragm, stop (a mechanical device in a camera that controls size of aperture of the lens) *"the new cameras adjust the diaphragm automatically"*
    - S: (n) finder, viewfinder, view finder (optical device that helps a user to find the target of interest)
    - S: (n) hood, lens hood (a tubular attachment used to keep stray light out of the lens of a camera)
    - S: (n) magazine, cartridge (a light-tight supply chamber holding the film and supplying it for exposure as required)
    - S: (n) shutter (a mechanical device on a camera that opens and closes to control the time of a photographic exposure)
    - S: (n) sprocket (roller that has teeth on the rims to pull film or paper through)

Further refinements

- Filter by dropping aspects that do not have sufficient mentions alongside known sentiment words (Blair-Goldensohn et al., 2008)
- Collapse aspects at the word stem level
- Pattern-based filter to remove non-aspect expressions (Moghaddam and Ester, 2010)
- Compare the frequencies of the candidates in a review corpus with those in a generic corpus to identify true aspects (Scaffidi et al., 2007)
- First find the core aspect words, then use the information distance to find other related words, e.g. "*$*" and "*dollars*" for "*price*" (Long et al., 2007)
- Etc., etc.

2. Exploiting syntactic relations

- Syntactic relations between sentiment expressions and their sentiment or opinion targets (Needs a parser)
  - E.g. "*This camera takes great photos.*"
  - "*Picture quality and battery life are great.*"
- Linguistic constructions
  - E.g. X of Y: "the *voice quality* of the *iPhone*"
  - Genitives: "the *camera*'s *price*"

Opinion-target relations

- Can use a dependency parser
  - Identify reliable dependency relation templates from training(labelled) data
  - Then use them to find valid aspect-sentiment pairs in test data
  - Can be used for simultaneous extraction of both sentiment words and opinion targets – Double Propagation (DP) method (Qiu et al. 2009, 2011)
  - E.g. "*The software is amazing*." -> pattern "NN – nsubj –JJ"

Tagging

```
The/DT  software/NN  is/VBZ  amazing/JJ  ./.
```

Parse

```
(ROOT
  (S
    (NP (DT The) (NN software))
    (VP (VBZ is)
      (ADJP (JJ amazing)))
    (. .)))
```

Universal dependencies, enhanced

```
det(software-2, The-1)
nsubj(amazing-4, software-2)
cop(amazing-4, is-3)
root(ROOT-0, amazing-4)
```

Stanford Parser
http://nlp.stanford.edu:8080/parser/index.jsp

Dependencies

- Dependencies are binary relations: a grammatical relation between
  - A *governor* (also known as *head*), and
  - A *dependent*
- For example, *amod* (adjectival modifier)
  - An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP.
  - E.g. *I like this amazing software.*

Universal Dependencies

	Nominals	Clauses	Modifier words	Function Words
Core arguments	nsubj obj iobj	csubj ccomp xcomp		
Non-core dependents	obl vocative expl dislocated	advcl	advmod* discourse	aux cop mark
Nominal dependents	nmod appos nummod	acl	amod	det clf case
Coordination	**MWE**	**Loose**	**Special**	**Other**
conj cc	fixed flat compound	list parataxis	orphan goeswith reparandum	punct root dep

Stanford Typed Dependencies

- *mod* – modifier

amod - adjectival modifier
appos - appositional modifier

advcl - adverbial clause modifier
det - determiner
predet - predeterminer
preconj - preconjunct
vmod - reduced, non-finite verbal modifier
mwe - multi-word expression modifier
    mark - marker (word introducing an *advcl* or *ccomp*
advmod - adverbial modifier
    neg - negation modifier
rcmod - relative clause modifier
quantmod - quantifier modifier

nn - noun compound modifier
npadvmod - noun phrase adverbial modifier
    tmod - temporal modifier
num - numeric modifier
number - element of compound number
prep - prepositional modifier
poss - possession modifier
possessive - possessive modifier (*'s*)
prt - phrasal verb particle

Stanford Typed Dependencies

- *arg* - arguments

  - *agent* - agent
  - *comp* - complement
    - *acomp* - adjectival complement
    - *ccomp* - clausal complement with internal subject
    - *xcomp* - clausal complement with external subject
    - *obj* - object
      - *dobj* - direct object
      - *iobj* - indirect object
      - *pobj* - object of preposition
  - *subj* - subject
    - *nsubj* - nominal subject
      - *nsubjpass* - passive nominal subject
    - *csubj* - clausal subject
      - *csubjpass* - passive clausal subject

Dependency Relations

- Common dependency relations between sentiment words and aspects include *amod*, *prep*, *nsubj*, *csubj*, *xsubj*, *dobj* and *iobj*
- Common relations for sentiment words and aspects themselves include: the conjunction relation *conj*

More examples

- "*The phone has a nice screen*."

Tagging

```
The/DT  phone/NN  has/VBZ  a/DT  nice/JJ  screen/NN  .
```

Universal dependencies, enhanced

```
det(phone-2, The-1)
nsubj(has-3, phone-2)
root(ROOT-0, has-3)
det(screen-6, a-4)
amod(screen-6, nice-5)
dobj(has-3, screen-6)
```

- "*I like the color of the phone*."

Universal dependencies, enhanced

Tagging

```
I/PRP  like/VBP  the/DT  color/NN  of/IN  the/DT  phone/NN
```

```
nsubj(like-2, I-1)
root(ROOT-0, like-2)
det(color-4, the-3)
dobj(like-2, color-4)
case(phone-7, of-5)
det(phone-7, the-6)
nmod:of(color-4, phone-7)
```

Example Rules for Aspect and Opinion Word Extraction

Rule ID	Observed Relation (Line 1) and Constraints (Lines 2–4)	Output	Examples
$R1_1$ (OA-Rel)	$O \rightarrow O\text{-}Dep \rightarrow A$ s.t. $O \in \{O\}$, $O\text{-}Dep \in \{MR\}$, $POS(A) \in \{NN\}$	$a = A$	The phone has a good "screen." $good \rightarrow mod \rightarrow screen$
$R1_2$ (OA-Rel)	$O \rightarrow O\text{-}Dep \rightarrow H \leftarrow A\text{-}Dep \leftarrow A$ s.t. $O \in \{O\}$, $O/A\text{-}Dep \in \{MR\}$, $POS(A) \in \{NN\}$	$a = A$	"iPod" is the best MP3 player. $best \rightarrow mod \rightarrow player \leftarrow subj \leftarrow iPod$
$R2_1$ (OA-Rel)	$O \rightarrow O\text{-}Dep \rightarrow A$ s.t. $A \in \{A\}$, $O\text{-}Dep \in \{MR\}$, $POS(O) \in \{JJ\}$	$o = O$	Same as $R1_1$ with *screen* as the known word and *good* as the extracted word
$R2_2$ (OA-Rel)	$O \rightarrow O\text{-}Dep \rightarrow H \leftarrow A\text{-}Dep \leftarrow A$ s.t. $A \in \{A\}$, $O/A\text{-}Dep \in \{MR\}$, $POS(O) \in \{JJ\}$	$o = O$	Same as $R1_2$ with *iPod* as the known word and *best* as the extract word
$R3_1$ (AA-Rel)	$A_{i(j)} \rightarrow A_{i(j)}\text{-}Dep \rightarrow A_{j(i)}$ s.t. $A_{j(i)} \in \{A\}$, $A_{i(j)}\text{-}Dep \in \{CONJ\}$, $POS(A_{i(j)}) \in \{NN\}$	$a = A_{i(j)}$	Does the player play DVDs with audio and "video"? $video \rightarrow conj \rightarrow audio$
$R3_2$ (AA-Rel)	$A_i \rightarrow A_i\text{-}Dep \rightarrow H \leftarrow A_j\text{-}Dep \leftarrow A_j$ s.t. $A_i \in \{A\}$, A_i-Dep=A_j-Dep OR (A_i-Dep = subj AND A_j-Dep = obj), $POS(A_j) \in \{NN\}$	$a = A_j$	Canon "G3" has a great lens. $len \rightarrow obj \rightarrow has \leftarrow subj \leftarrow G3$
$R4_1$ (OO-Rel)	$O_{i(j)} \rightarrow O_{i(j)}\text{-}Dep \rightarrow O_{j(i)}$ s.t. $O_{j(i)} \in \{O\}$, $O_{i(j)}\text{-}Dep \in \{CONJ\}$, $POS(O_{i(j)}) \in \{JJ\}$	$o = O_{i(j)}$	The camera is amazing and "easy" to use. $easy \rightarrow conj \rightarrow amazing$
$R4_2$ (OO-Rel)	$O_i \rightarrow O_i\text{-}Dep \rightarrow H \leftarrow O_j\text{-}Dep \leftarrow O_j$ s.t. $O_i \in \{O\}$, O_i-Dep=O_j-Dep OR (O_i /O_j-Dep $\in$ {pnmod, mod}), $POS(O_j) \in \{JJ\}$	$o = O_j$	If you want to buy a sexy, "cool," accessory-available MP3 player, you can choose iPod. $sexy \rightarrow mod \rightarrow player \leftarrow mod \leftarrow cool$

Lexico-syntactic patterns

- Genitive constructions are frequently used to express part-of and attribute-of relations.
  - "*the battery of the iPhone*" – "NP-head of NP-mod"
  - "*the iPhone's sound quality*" – "NP-mod's NP-head"
- However, the semantic relations of the two nouns can be quite different in different context
  - **Part-of**: "*iPhone's battery*"
  - Possession: "*John's iPhone*"
  - **Attribute-of**: "*iPhone's price*"
  - Kinship: "*John's brother*"
  - Source-from: "*John's birth city*"
  - Make-produce: "*Apple's phone*"
- In sentiment analysis, it's easier to **fix NP-mod to be a specific entity to look for its aspect**

The approach with no parser

- Full parsing is expensive (typically fewer than 20 sentences per second), and doesn’t work well on informal data.
- **Approximate** the dependency using distance:
  - If a sentence does not have a frequent aspect but has some sentiment words, the nearest noun or noun phrase to a sentiment word is extracted as an aspect.
- **Approximate** the dependency relations
  - Using linear patterns of words and POS tags, or chunk patterns (from shallow parsing)
  - And extract using a good pattern matching algorithm.
- Very useful in practice.

Ranking the candidates

- Aspect relevance (*r(a)*): how likely it's a genuine aspect
  - If an aspect is modified by multiple sentiment words
    "*delivery*" modified by "*quick*", "*cumbersome*", "*timely*"
  - If it's extracted by multiple lexico-syntactic patterns
    - "*The engine of the car is large*" and "*The car has a big engine*"
  - If it's extracted by both a sentiment word modification relation and a lexico-syntactic pattern in the same sentence
    - "*There is a bad hole in the mattress*"
- Aspect frequency (*f(a)*)
  - Rank the frequent aspects higher
- Final ranking score: $S(a) = r(a) \log(f(a))$

Going further

- Use phrase dependency parser to extract noun phrases and verb phrases.
- Adding comparative- and superlative-based relations
  - E.g. "*The iPhone 5 has better voice quality than Moto X*."
- Adding sentiment composition rules
  - E.g. "*Enbrel has reduced my joint pain*."
- Resource usage aspect - resource expression, (*usage_verb, quantifier, resource_noun*)
  - E.g. "*This washer uses a lot of water*."

  usage

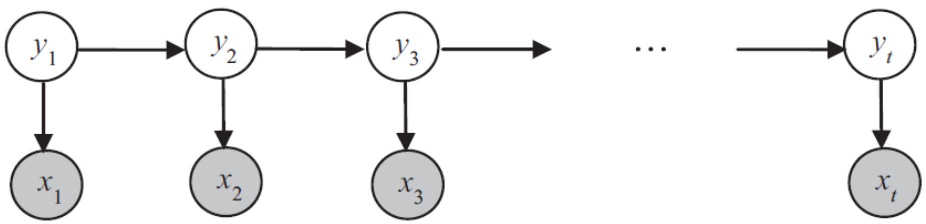
3. Supervised learning

- Treat aspect extraction as a special IE problem.
- Dominant method: sequential learning or sequence labeling like *hidden Markov models* (HMMs) and *conditional random fields* (CRF).
- Many other methods, like sequential rules, tree-structured classification, etc. have been tried as well.
- Requires labelled data.

HMM

- Directed sequence model, successfully applied to many sequence labeling problems such as NER and POS tagging

$\mathbf{y} = < y_0, y_1, \ldots y_t >$: hidden state sequence
$\mathbf{x} = < x_0, x_1, \ldots x_t >$: observation sequence



- Assumptions:
  - state *yi* only depends on its immediate predecessor state *yi*-1 (Markov Assumption)
  - the observation *xi* only depends on the current state *yi*.
- For aspect extraction,
  - Observations: words or phrases in a review
  - Underlying states: aspect or opinion expression tags

HMM

- The **joint probability** of a state sequence **y** and an observation sequence **x**

$$p(\mathbf{y}, \mathbf{x}) = \prod_{i=1}^{t} p(y_i | y_{i-1}) p(x_i | y_i)$$

  - $p(y_0)$ over the initial state,
  - a state transition distribution $p(y_i|y_{i-1})$,
  - an observation distribution $p(x_i|y_i)$

- Learning the model: given some observation sequences, learn the model parameter that maximizes the observation probability, e.g. MLE from counts.

- Applying the model (decoding): find an optimal state sequence for a new observation sequence, e.g. Viterbi algorithm

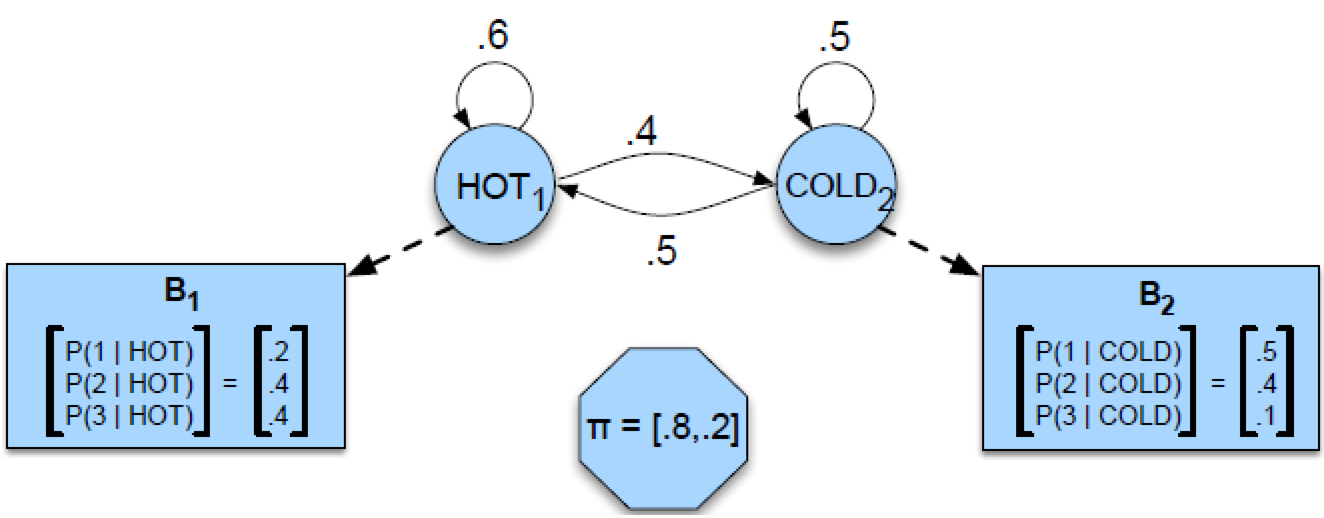
E.g. Lexicalized HMM

- Given a sequence of words $W = w_1w_2w_3\ldots w_n$ and corresponding parts-of-speech $S = s_1s_2s_3\ldots s_n$,
- The task is to find an appropriate sequence of tags $T = t_1t_2t_3\ldots t_n$ that maximize the conditional probability $P(T|W,S)$
- For example: "*I love the ease of transferring the pictures to my computer*."
- Tags:

<BG>I</BG><OPINION_POS_EXP>love</OPINION_POS_EXP><BG>the</BG><PROD_FEAT-BOE>ease</PROD_FEAT-BOE> <PROD_FEAT-MOE> of</PROD_FEAT-MOE><PROD_FEAT-MOE>transferring</PROD_FEAT-MOE> <PROD_FEAT-MOE>the</PROD_FEAT-MOE> <PROD_FEAT-EOE>pictures</PROD_FEAT-EOE> <BG>to</BG><BG>my</BG><BG>computer</BG>

Understanding HMM

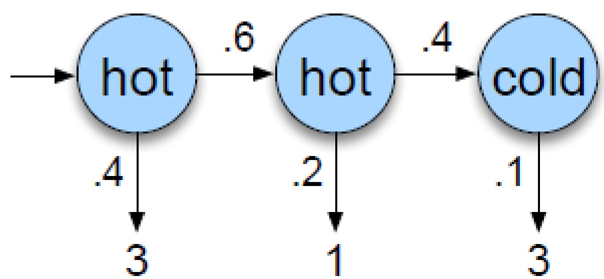
- Hidden states: H(ot) and C(old) weather
- Observations: the number of ice creams eaten on that day – {1, 2, 3}
- Transition probabilities
- Emission probabilities



Sequence probability

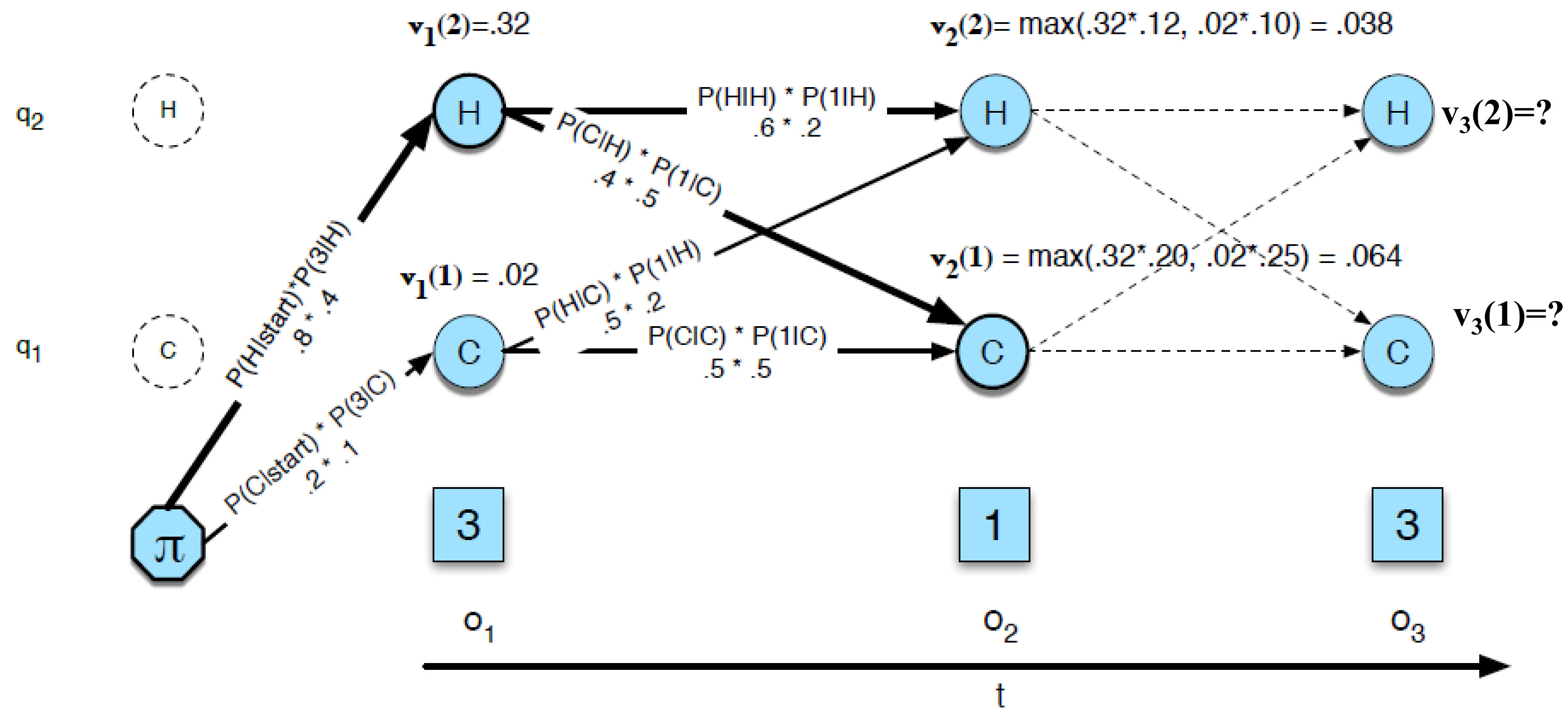
- E.g. to compute the joint probability of observation sequence <3, 1, 3> and hidden state sequence <hot, hot, cold>

$$
\begin{aligned}
P(3\ 1\ 3, \text{hot hot cold}) &= P(\text{hot}|\text{start}) \times P(\text{hot}|\text{hot}) \times P(\text{cold}|\text{hot}) \\
&\quad \times P(3|\text{hot}) \times P(1|\text{hot}) \times P(3|\text{cold})
\end{aligned}
$$



Viterbi Decoding

- Given the observation sequence <3, 1, 3>, find the best sequence of hidden states



Exercise

- The previous slide shows the computation of $v_t(j)$ for two states at two time steps: $v_1(1)$ & $v_1(2)$, $v_2(1)$ & $v_2(2)$

$$v_t(j) = \max_{1 \leq i \leq N-1} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

- Now try to compute the values for $v_3(1)$ & $v_3(2)$

Conditional Random Fields

- CRF is a discriminative classifier model (HMM is a generative model)
- Extension of logistic regression to sequential data, easier to incorporate a lot of features
- Undirected graphical model, thus its features are not restricted to tags of the preceding nodes -> bidirectional
- It models a conditional probability p(**y**|**x**) over hidden sequence **y** given observation sequence **x**
- Trained to label an unknown observation sequence **x** by selecting the hidden sequence **y** that maximizes p(y|x).

CRF in NLP context

- Suppose (X, Y) is a conditional random field such that X are the '*observables*' and Y is a '*latent' variable*. In NLP, Y can be the EA tags (that you want to learn) while X could be the POS tags.

$$p(\mathbf{y}|\mathbf{x}) = \underbrace{\frac{1}{Z(\mathbf{x})}}_{\text{Normalization}} \prod_{t=1}^{T} \exp\left\{\sum_{k=1}^{K} \underbrace{\theta_k}_{\text{Weight}} \underbrace{f_k(y_t, y_{t-1}, \mathbf{x}_t)}_{\text{Feature}}\right\}$$

- E.g. the EA tag of a word can be determined by the word, its POS tag, its neighbouring words with their respective POS and EA tags
- The expression is a conditional probability which is computed by Bayes rule statistically from the learning corpus and filters out the most likely sequence of X and Y.

Mathematical intuition - CRF

	the	only	redeeming	factor	was	the	food
X	DT	JJ	NN	NN	VBD	DT	NN
Y	O	O	O	O	O	O	B-A

- The CRF algorithm computes probabilities for eg.:
  - P(I-A | I-A, NN)
  - P(B-A| I-A, NN) … …and even ‘emission’ probabilities from the P(NN| O) (from unobserved to observed)

- Based on the corpus with labelled sequences, the most or least likely ones will then be determined.

B-A: beginning of aspect
I-A: part of aspect
O: anything out of vocabulary

Implicit aspect

- Explicit aspects
  - "*The picture quality of the camera is great.*"
- Implicit aspects - opinion expressions that do not mention aspect explicitly as a noun

  "*The camera is expensive*" ("price")

  "*This beautiful camera*…" ("appearance")

  "*The camera does not easily fit in a pocket*" ("size")
- Mapping adjectives or verb phrases to aspects
  - Corpus-based approach: using the co-occurrence of sentiment words and explicit mentions e.g. "*The size of the phone is small*" vs "*the phone is small*"
  - Dictionary-based approach: using dictionary definition of "*expensive*" – "*marked by high prices*"

Grouping aspects into categories

- People use different words or phrases to describe the same aspect or aspect category.
  e.g. "*sound quality*", "*voice quality*"
- Need to group **aspect expressions** into **aspect categories**, each representing one aspect
- Very challenging as it's subjective task. Different application or different users may require different categories based on application need or granularity of analysis
- Can use WordNet or other thesaurus to find synonyms, but not sufficient
  - Domain dependent synonyms
  - Multiword phrases, not in WordNet or dictionaries
  - Not synonyms – "expensive" and "cheap"

Some methods

- Aspects sharing common words: "*battery life*" and "*battery power*"
- Aspects that are synonyms in dictionaries: "*movie*" and "*film*"
- Aspects with short lexical distances measured in WordNet: "*movie*" and "*show*"
- Mapping aspect expressions to aspect nodes in an existing taxonomy, based on similarity
- Using topic modelling (LDA)
- Other semi-supervised methods

RECENT ADVANCES IN NLP (AND SENTIMENT ANALYSIS)

Pre-2012

- Bag of words approach
  - does not consider the order (semantics nor context) of words:
  - Term Document Matrix, tf-idf indexing, etc.
- Lots of work at document classification, like SVC, MaxEnt, NB, etc.
- Some attempts to get to the semantics, like Latent semantic indexing
- Incremental improvement by feature engineering (still useful)

2013 to 2016 – Word Embeddings

- Vectorization using word embedding
  - A vector representation of a word (word vectors) from contextual corpus (map a word using a corpus into a n-dimensional vector)
  - Word2Vec (2013 by Google) or GLOVE (2014 by Stanford)
- Task:
  - Word2vec - using the word itself to predict its probable surrounding words.
  - GLOVE - using the contextual *window* words to predict the word itself.
- Resulting in embeddings that somehow capture the semantics of words

Deep Learning

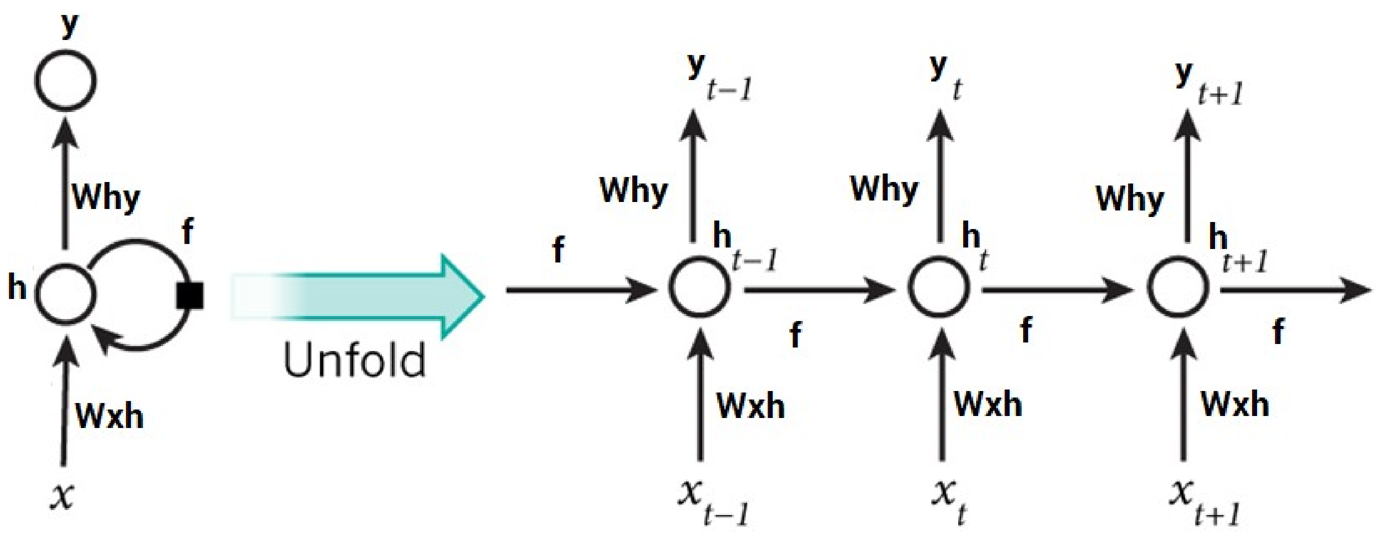
- The latest techniques in NLP come from Deep Learning.
  - FNN – feedforward neural network
  - CNN – convolutional neural network
  - RNN – recurrent neural network (Sequence modelling!)

- This will be covered in greater detail in the next course - TPML

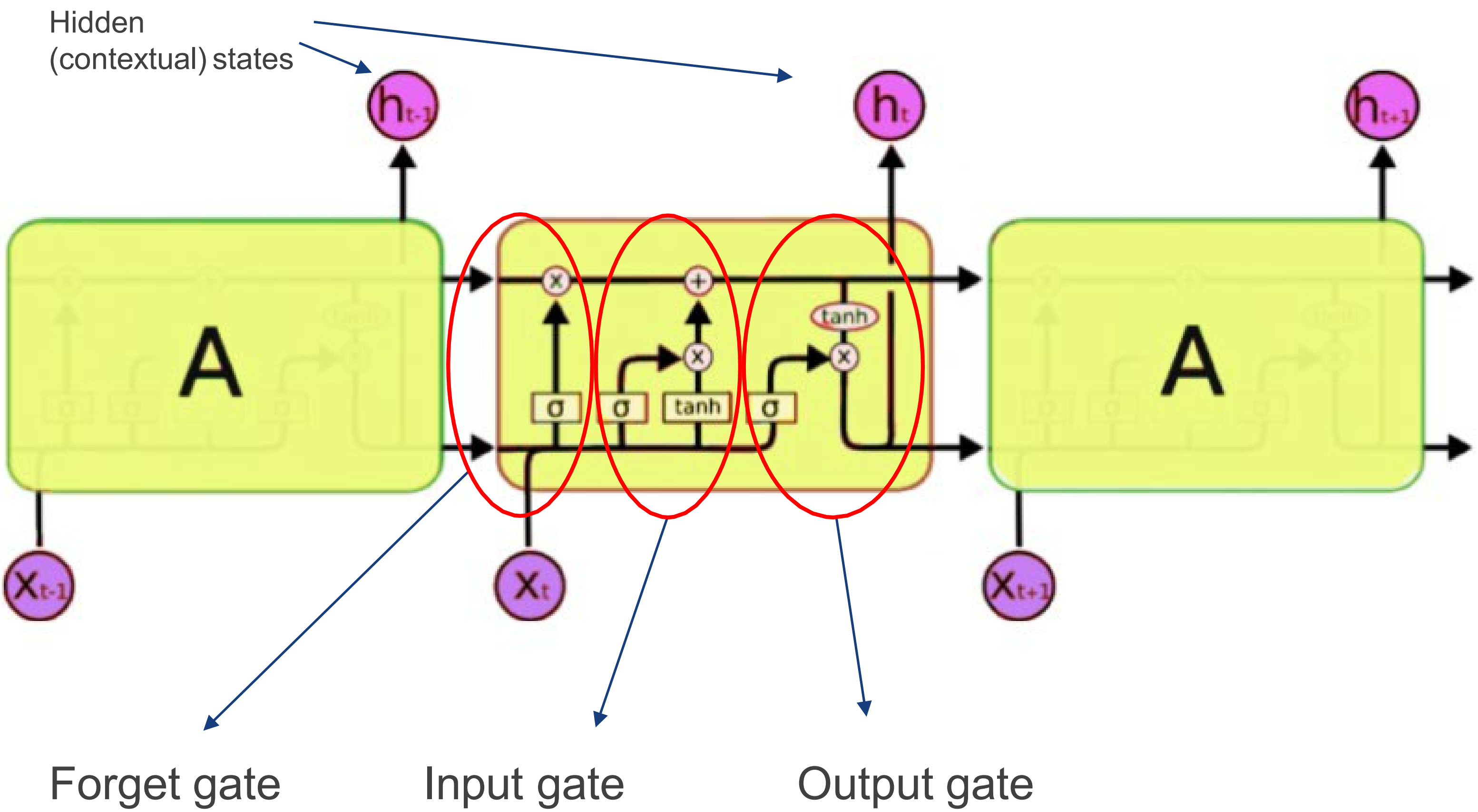
Recurrent Neural Network

- This is commonly used for NLP.
- Learn through sequential dependence of x_t (eg time series data, language sequences)
- Popular improved variant eg. LSTM (long-short term memory)

$$f(x_{1t}, x_{2t}, x_{3t} \ldots) \rightarrow y_{1t}, y_{2t}, y_{3t}$$



Long-Short Term Memory



LSTM 'remedies' RNN problems of vanishing gradients (due to the long sequence) by having a sigmoidal 'forget' gate that chooses whether to include the h_{t-n} as either 1 or 0.

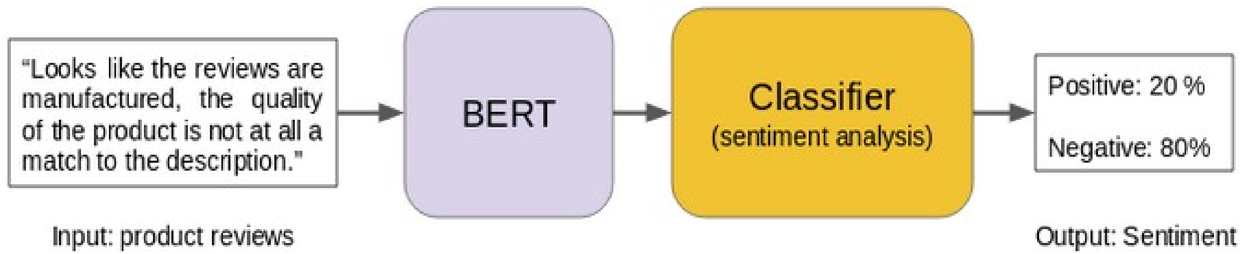
Transformers with Attention

- Many NLP tasks can benefit from such sequence modelling. However there are still issues
  - Vanishing gradients
  - Cannot parallelize
- Solution: **Transformer**
  - A novel architecture of ML using attention (focuses on the features that really matter)
  - The ‘transformer’ here is an encoding/ decoding layer with a special attention layer
  - Better handling of long term dependencies

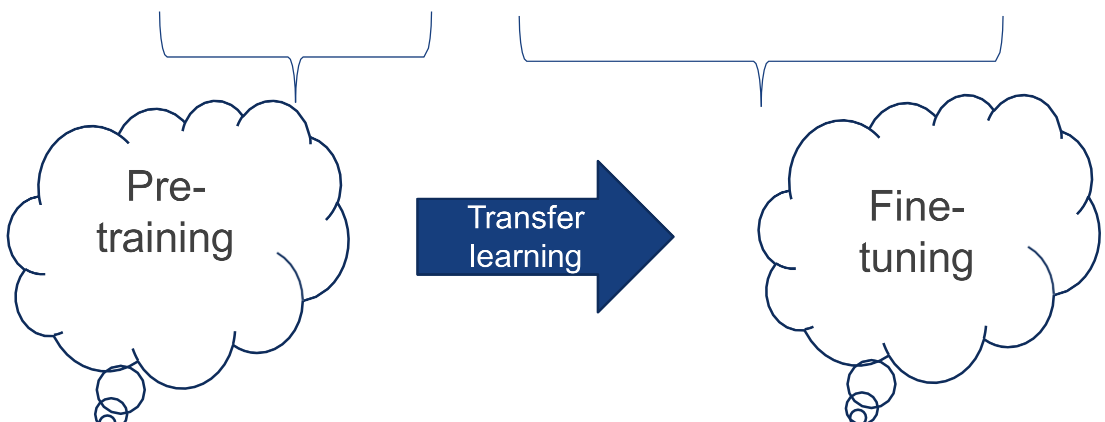
Huge advances from 2018

- Elmo (Mar 2018) – contextualized word embeddings
- ***Pre-trained*** models
    - Pre-trained transformers using huge amount of text data
    - The pre-training helps to capture context, syntactic and semantic information in the text.
    - Can be further fine-tuned using small data set to perform a wide variety of NLP tasks, including sentiment mining tasks
    - BERT, GPT/GPT2/GPT3, XLNet, T5, BART, etc.

BERT for sentiment analysis



Fine-Tuning BERT on a Sentiment Analysis Task



By *Google* –

- BERT base – no of transformer layers = 12; total parameters = 110M
- BERT Large: Number of Transformers layers = 24, Total Parameters = 340M

Sentiment analysis, text classification (eg. spam filters, toxic comments, Q&A)

Leading from the start

Rank	Model	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	QNLI	RTE
1	BERT: 24-layers, 1024-hidden, 16-heads	80.4	60.5	94.9	85.4/89.3	87.6/86.5	89.3/72.1	86.7	91.1	70.1
2	Singletask Pretrain Transformer	72.8	45.4	91.3	75.7/82.3	82.0/80.0	88.5/70.3	82.1	88.1	56.0
3	BiLSTM+ELMo+Attn	70.5	36.0	90.4	77.9/84.9	75.1/73.3	84.7/64.8	76.4	79.9	56.8

A few years back from
https://gluebenchmark.com

A race is going on…

Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI	RTE	WNLI
1	AliceMind & DIRL	StructBERT + CLEVER		91.0	75.3	97.7	93.9/91.9	93.5/93.1	75.6/90.8	91.7	91.5	97.4	92.5	95.2
2	ERNIE Team - Baidu	ERNIE		90.9	74.4	97.8	93.9/91.8	93.0/92.6	75.2/90.9	91.9	91.4	97.3	92.0	95.9
3	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4		90.8	71.5	97.5	94.0/92.0	92.9/92.6	76.2/90.8	91.9	91.6	99.2	93.2	94.5
4	HFL iFLYTEK	MacALBERT + DKM		90.7	74.8	97.0	94.5/92.6	92.8/92.6	74.7/90.6	91.3	91.1	97.8	92.0	94.5
+ 5	PING-AN Omni-Sinitic	ALBERT + DAAF + NAS		90.6	73.5	97.2	94.0/92.0	93.0/92.4	76.1/91.0	91.6	91.3	97.5	91.7	94.5
6	liangzhu ge	Deberta + adv (ensemble)		90.4	72.7	97.3	92.7/90.3	93.2/92.9	75.6/90.8	91.7	91.5	96.4	92.5	95.2
7	T5 Team - Google	T5		90.3	71.6	97.5	92.8/90.4	93.1/92.8	75.1/90.6	92.2	91.9	96.9	92.8	94.5
8	Microsoft D365 AI & MSR AI & GATECH	MT-DNN-SMART		89.9	69.5	97.5	93.7/91.6	92.9/92.5	73.9/90.2	91.0	90.8	99.2	89.7	94.5
+ 9	Huawei Noah's Ark Lab	NEZHA-Large		89.8	71.7	97.3	93.3/91.0	92.4/91.9	75.2/90.7	91.5	91.3	96.2	90.3	94.5
+ 10	Zihang Dai	Funnel-Transformer (Ensemble B10-10-10H1024)		89.7	70.5	97.5	93.4/91.2	92.6/92.3	75.4/90.7	91.4	91.1	95.8	90.0	94.5

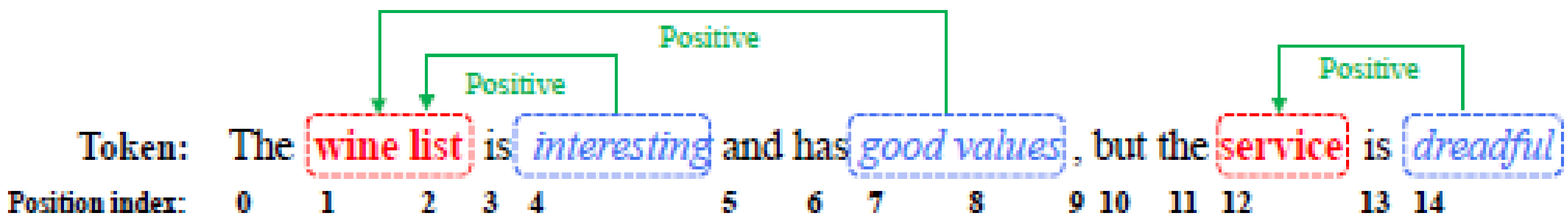
https://gluebenchmark.com/leaderboard

ABSA with DL

- Most treat extraction as a sequence tagging task, using LSTM-based models, or pre-trained transformers.
- An example solution using BART
  - a strong sequence-to-sequence pre-trained model for natural language generation.
  - Encoder and decoder layers
  - Pre-training task: take a masked or permuted sentence as input, return the restored sentence.
- Given input text, generate a sequence of token indexes and class indexes for various ABSA tasks

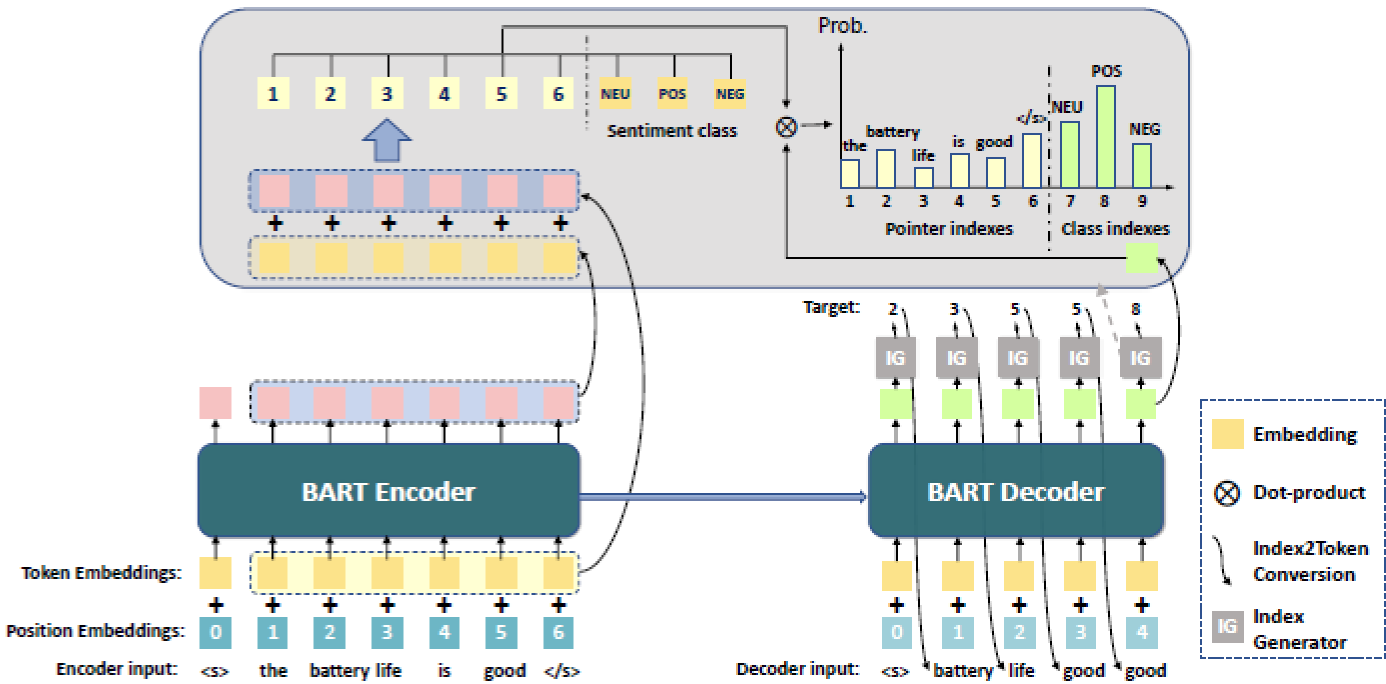
Yan, Hang, et al. "A Unified Generative Framework for Aspect-Based Sentiment Analysis." *arXiv preprint arXiv:2106.04300* (2021).

ABSA subtask formulation



Subtask	Target Sequence
AE	1, 2, 12, 12, </s>
OE	4, 4, 7, 8, 14, 14, </s>
ALSC	1, 2 , POS, </s>
	12, 12, POS, </s>
AOE	1, 2, 4, 4, 7, 8, </s>
	12, 12, 14, 14, </s>
AESC	1, 2, POS, 12, 12, NEG, </s>
Pair	1, 2, 4, 4, 1, 2, 7, 8, 12, 12, 14, 14, </s>
Triplet	1, 2, 4, 4, POS, 1, 2, 7, 8, POS, 12, 12, 14, 14, POS, </s>

Overall architecture



Source sequence: “<s> the battery life is good </s>”

Target sequence: “2 3 5 5 8 6”

References

- Liu, Bing. ”**Chapter 6: Aspect and Entity Extraction**”. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 2015.
- Liu, Bing. "Sentiment analysis and opinion mining." *Synthesis Lectures on Human Language Technologies* 5.1 (2012): 1-167.
- Pang, Bo, and Lillian Lee. “Opinion Mining and Sentiment Analysis.” *Foundations and Trends in Information Retrieval* 2.1-2 (2008): 1-135.
- Stanford Typed Dependencies Manual (https://nlp.stanford.edu/software/dependencies_manual.pdf)
- Universal Dependencies (https://universaldependencies.org/u/dep/all.html/)