"Life Stories" and text data mining

Mark Johnson
Lan Du, Anish Kumar
Wray Buntine (Monash)
Massimiliano Ciaramita (Google Research)

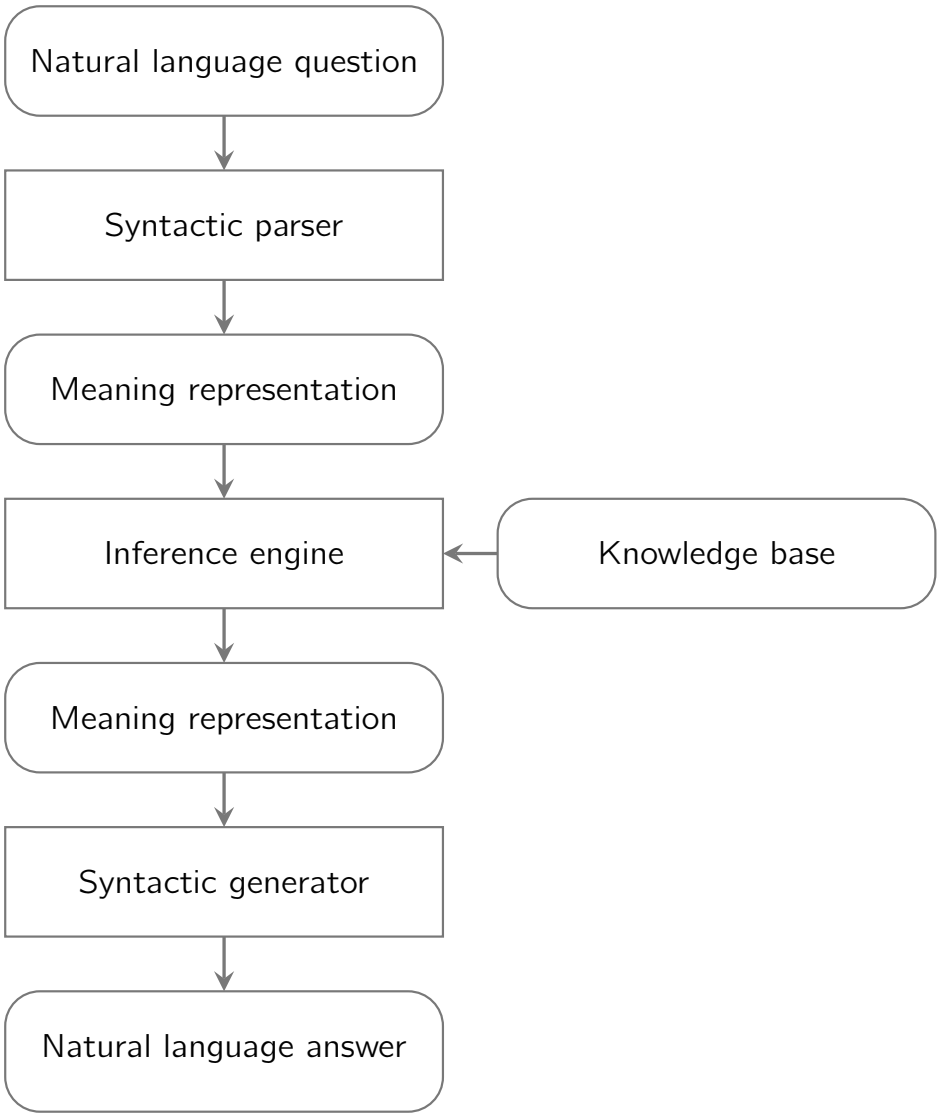
Macquarie University
Sydney, Australia

May 2014

MACQUARIE UNIVERSITY

Outline

GofAI (Good old-fashioned AI)



- Good "old fashioned" AI approaches to Natural Language Processing (NLP) circa 1980s–1990s failed because:
  - syntactic parsers and generators were inaccurate and not robust
  - *knowledge bases were hopelessly incomplete*

MACQUARIE UNIVERSITY

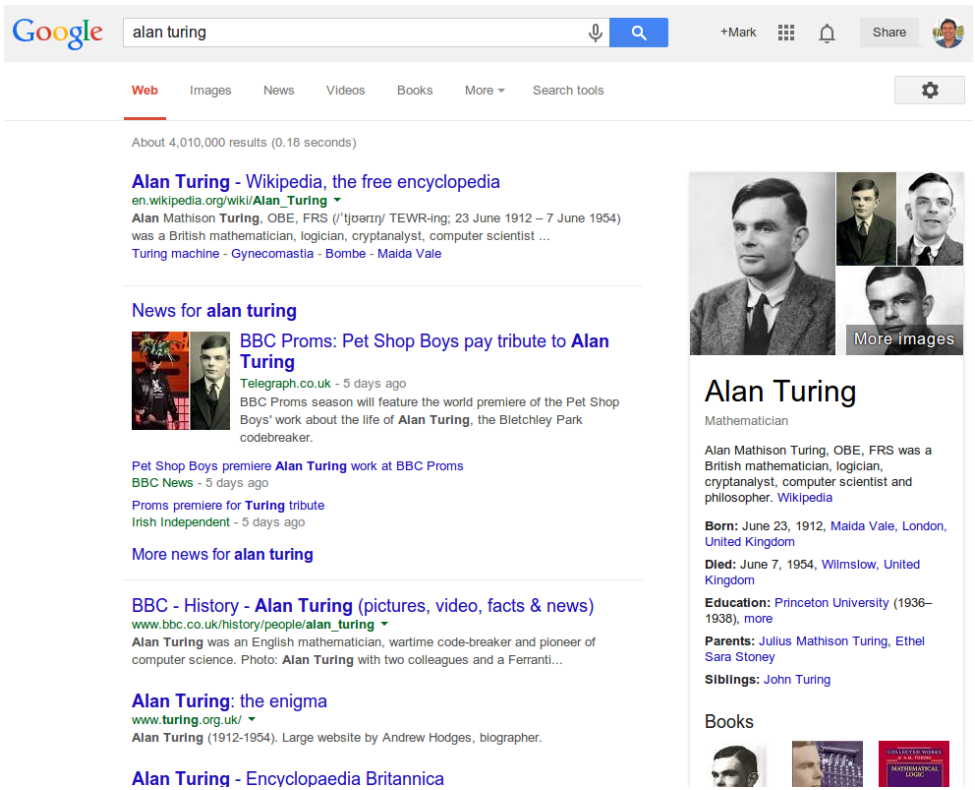
Advantages of probabilistic models in NLP

- In the GofAI approach to syntactic parsing:
  - a hand-written grammar defines the grammatical (i.e., well-formed) parses
  - given a sentence, the parser returns the set of grammatical parses for that sentence

  $\Rightarrow$ unable to distinguish more likely from less likely parses

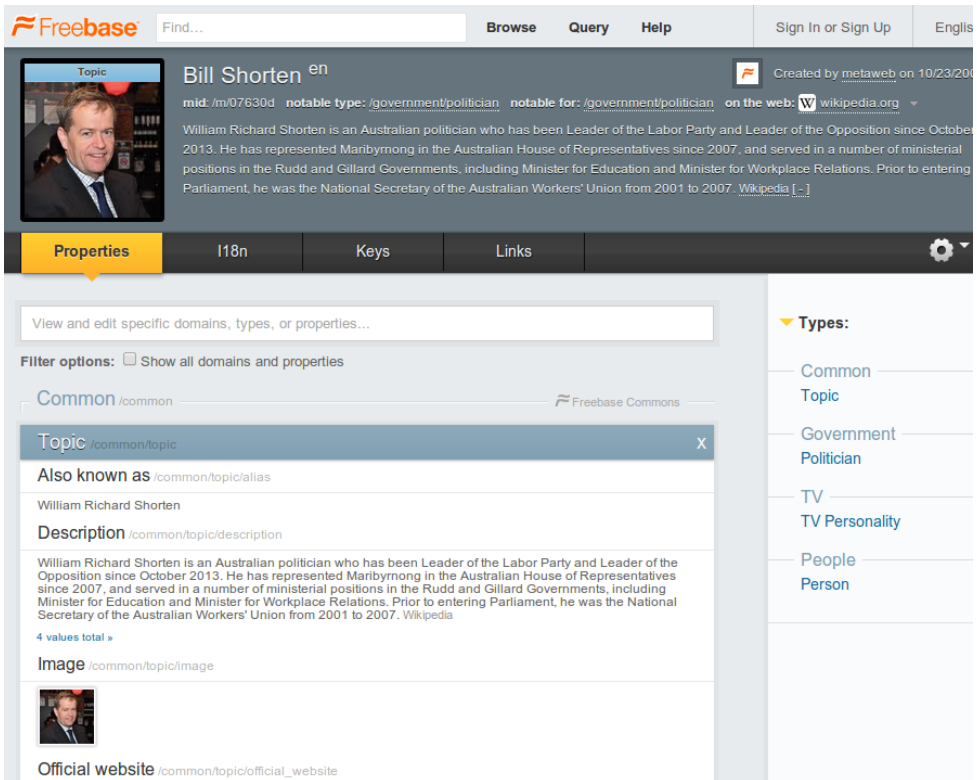
  $\Rightarrow$ hard to ensure *robustness* (i.e., that every sentence gets a parse)
- In a probabilistic parser:
  - the grammar *generates all possible parse trees* for all possible strings (roughly)
  - use probabilities to identify plausible syntactic parses
- Probabilistic syntactic models usually encode:
  - the probabilities of syntactic constructions
  - the probabilities of lexical dependencies
    e.g., how likely is *pizza* as direct object of *eat*?

Google's Knowledge Graph



- Goal: move beyond keyword search document retrieval to *directly answer user queries*
  - ⇒ easier for mobile device users
- Google's Knowledge Graph:
  - built on top of FreeBase
  - entries are synthesised from Wikipedia, news stories, etc.
  - manually curated (?)

FreeBase: an open (?) knowledge base



- An entity-relationship database on top of a graph triple store
- Data mined from Wikipedia, ChefMoz, NNDB, FMD, MusicBrainz, etc.
- 44 million topics (entities), 2 billion facts, 25GB compressed dump
- Created by Metaweb, which was acquired by Google

MACQUARIE UNIVERSITY

Outline

The lay of the land

- Several overlapping fields using similiar techniques:
  - *Statistics:* summarisation of and hypothesis testing with quantitative data
  - *Machine learning:* predicting properties of novel data (artificial intelligence, computer science)
  - *Data mining:* summarising and identifying regularities in data sets (computer science, business)
    - *Text data mining:* information extraction from document collections (information retrieval, computer science)
  - *Analytics:* modelling and displaying quantitative data (statistics, business)
  - *Big data:* algorithms/hardware for large data sets (computer science)
  - *Data science:* cover term for all the above
- *Probability and statistics* play a foundational role in all of these

Machine learning and statistics

- Probabilistic models are basis of statistics and machine learning
  - *probability theory* models *randomness* and *uncertainty*
  - a *statistic* is a function of the data (designed to summarise it)
- Statistics is primarily concerned with *hypothesis testing*
  - statistical tests identify which *predictors* are *significantly associated* with the *predicted variable*
  - e.g., *does coffee cause cancer?*
  - a big model may have tens of predictors
- Machine learning is primarily concerned with *prediction*
  - models map *data items* to predicted variables or *labels*
  - *learn from training data* to predict labels of novel data items
  - e.g., *who is likely to get cancer?*
  - a big model may have millions of predictors or *features*
  - in general we can't tell which predictors are significant
  - but by *pooling the predictors* can often make accurate predictions

Example machine learning classification task

Training data:

Features	Label
Gender=male, Age=27, CoffeeDrinker=yes, MeatEater=no	Disease=cancer
Gender=female, Age=22, CoffeeDrinker=yes, MeatEater=no	Disease=none
Gender=male, Age=22, CoffeeDrinker=no, MeatEater=no	Disease=none
Gender=male, Age=18, CoffeeDrinker=yes, MeatEater=yes	Disease=none

Testing data:

Features	Label
Gender=male, Age=24, CoffeeDrinker=yes, MeatEater=no	Disease=?
Gender=female, Age=32, CoffeeDrinker=no, MeatEater=no	Disease=?

A typology of machine learning problems

	Supervised learning	**Unsupervised learning**
Discrete labels	Classification	Clustering
Continuous labels	Regression	Dimensionality reduction

- Supervised vs. unsupervised learning:
  - supervised: training data contains labels for data items
  - unsupervised: training data does not contain labels for data items
- Continuous vs. discrete labels:
  - continuous labels: labels are (vectors of) real numbers
  - discrete labels: labels come from a countable set

Structured prediction problems

- In some problems the labels have a *complex structure*
  - in *natural language parsing* the data items are sentences and the labels are syntactic parse trees
  - in *image analysis* the data items are images (pixel arrays) and the labels are mappings from pixels to object identifiers
  - in *protein binding site prediction* the data items are nucleotide sequences and the labels are mappings from the nucleotides to booleans (active/inactive)
- Often there are *structural constraints that the labels must satisfy*
  - in natural language syntactic parsing, the label must be a well-formed tree
- Often the *number of possible labels is astronomical*
  - a grammar we use for natural language parsing generates more than 10^{70} trees for some fairly ordinary sentences

$\Rightarrow$ Finding the optimal label can be computationally difficult

Training a machine learning model

- Models usually have a large number of *parameters*
  - one or more parameters per feature indicating e.g., the importance of that feature
  - parameter values are estimated, *learned* or *trained* from *training data*
- *Training data* consists of a large number of instances pairing sets of features and labels
- Parameter estimation or *training* is an *optimisation problem*
  - first version: find parameter values that *minimize* the *prediction errors* that the model makes on the training data

$\Rightarrow$ Requires optimising functions with tens of millions of parameters over tens of millions of data items

  - typically computationally most intensive part of machine learning
  - standard numerical optimisation techniques can sometimes be used
  - *on-line algorithms* only make one pass over the training data

Outline

Named entity recognition and linking

- *Named entity recognition* finds all "mentions" referring to an entity in a document

  Example: *Tony Abbott* *bought* *300* *shares in* *Acme Corp* *in* *2006*

  person, number, corporation, date

- *Noun phrase coreference* tracks mentions to entities within or across documents

  Example: *Julia Gillard met the president of Indonesia yesterday. Ms. Gillard told him that she . . .*

- *Entity linking* maps entities to database entries

  Example: *Tony Abbott* *bought* *300* *shares in* *Acme Corp* *in* *2006*

  /m/xw2135, number, /m/yzw9w, date

Relation extraction

- *Relation extraction* mines texts to find *relationships between named entities*, i.e., "who did what to whom (when)?"

  The new Governor General, Peter Cosgrove, visited Buckingham Palace yesterday.

Has-role

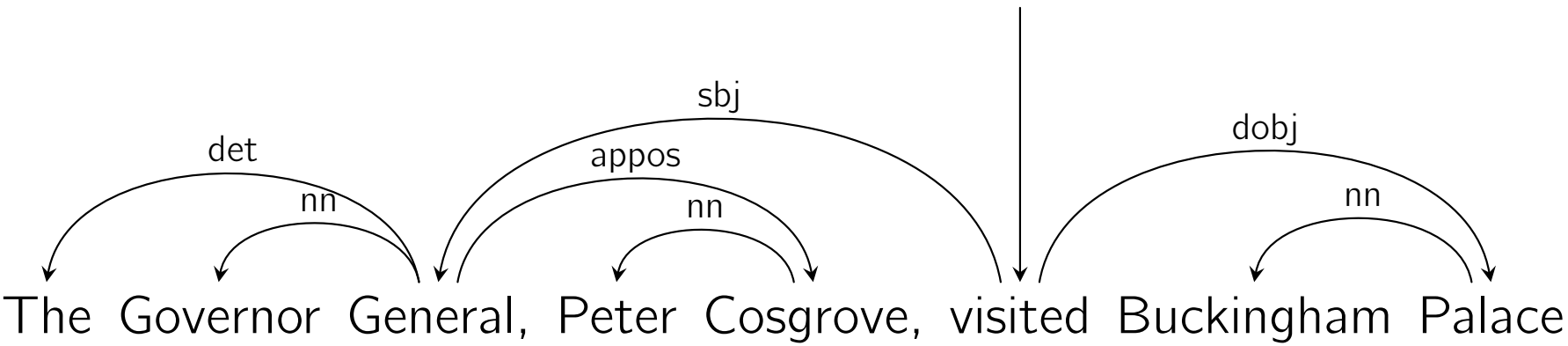
Person	Role
Peter Cosgrove	Governor General of Australia

Offical-visit

Visitor	Organisation
Peter Cosgrove	Queen of England

- Used to extend Knowledge Graph and FreeBase
- Text mining bio-medical literature is a major application

Syntactic parsing is useful in relation extraction



- The *syntactic path* in a *dependency parse* is a useful feature in relation extraction

$$X < \text{appos} < Y \Rightarrow \textit{has-role}(Y, X)$$

$$X > \text{sbj} > \textit{visited} < \text{dobj} < Y \Rightarrow \textit{official-visit}(X, Y)$$

Outline

Which Jim Jones?

- News text: *Jim Jones' recent musical releases . . .*
- 8 Wikipedia pages for *Jim Jones*:
  - 2 politicians
  - 1 basketball player
  - 1 hockey player
  - 1 guitarist (deceased)
  - 1 rapper
  - 1 cult leader (deceased)
- *How do we know it's the rapper?*

Life Stories

- A person's *life story* is the sequence of events that occur to them
- Generalisations about life stories:
  - everyone dies less than 110 years after they were born
  - if someone goes to school,
    it's usually when they are 5–20 years old
  - if someone goes to college,
    it's often immediately after school
  - a singer is more likely than a carpenter
    to have a musical release
  - an academic is more likely than an accountant
    to write a book
  - a lawyer is more likely than an actor
    to become a politician

The structure of life stories

- Everybody's life story is different
  - $\Rightarrow$ finite set of "life templates" won't suffice
- But there are generalisations:
  - few artists have exactly 10 CDs like Jim Jones
  - but releasing a CD is a frequent event for artists like Jim Jones, with predictable subevents:
    - release parties
    - promotions and reviews
    - shows and tours
- *Can we learn typical life stories?*
- *Given a partial life story, can we "fill in" the rest?*

Life Stories and Topic Models

LDA topic models	Life story models
words	*events* (e.g., running for election, releasing a CD)
documents	*life stories* (the sequence of events in an individual's life)
topics	*careers* (sequences of events associated with e.g., being a politician or musician)

- Topics are hidden when training a topic model, while FreeBase has abundant information about events
  - identifying the *relevant information* may be hard

What are Life Stories?

- FreeBase as a repository of Life Stories
  - FreeBase contains more than 100 properties for $\approx$ 250,000 people
  - Coverage is uneven: Sarah Palin's political career is covered, her political commentator roles on Fox News are not
- What appears in a Life Story?
  - time-stamped properties, e.g., *Bill Clinton's presidency 1993–2001*
  - indirectly time-stamped properties, e.g., *Bill Clinton's 1996 presidential campaign*
  - some properties without timestamps, e.g., *gender, nationality, notable type*

Important events

- Events differ in importance
  - Bill Clinton made 97 political appointments,
    appeared on 24 TV shows, and
    was elected US President twice
- Knowledge-base internal measures of importance (?)
  - *causes* are highly predictive, temporally-preceding event types (?)
- External measures of importance or impact
  - use relation extraction to align FreeBase properties to the individual's Wikipedia text, or a large news corpus
  - estimate importance by *amount of text* (sentences, column inches, etc.) linked to event

Event structure

- Events have a complicated *temporal* and *causal* structure
  - Bill Clinton's winning the 1996 Presidental election

  $\Rightarrow$ Bill Clinton is US President 1997–2001

  $\Rightarrow$ Bill Clinton makes 97 political appointments
- At what *granularity* should we individuate events?
  Many useful tasks don't require detailed information
  - dead cult leaders don't release hit CDs
- Minor events can give information about important events
  - a late alimony payment $\Rightarrow$ marriage and divorce
- Can *hierarchical models* generalise at multiple levels simultaneously?

Evaluating a Life Story model

- Life Story models should be useful in
  - named entity linking
  - relation extraction

  but accuracy on those tasks depends on other factors as well
- Evaluate the predictive ability of a Life Story model, e.g.:
  - train model on 2012 FreeBase
  - give model an individual's pre-2013 Life Story and several possible 2013 completions
  - evaluate how accurately model chooses correct completion

Example: Dick Cheney

The story until 2000

- born 1941, in Lincoln, Nebraska
- studied political science at the University of Nebraska
- White House chief of staff 1975–1977
- elected to US Congress 1979–1989
- minority whip in US Congress 1989
- US Secretary for Defense 1989–1993
- employed by Halliburton 1995–2000

2001 alternative #1

- litigant in Supreme Court legal case
- Vice President of the United States
- founded Energy Task Force

2001 alternative #2

- mayor of Wasilla, Alaska
- member of the Alaska Municipal League board

Some possible Life Story models

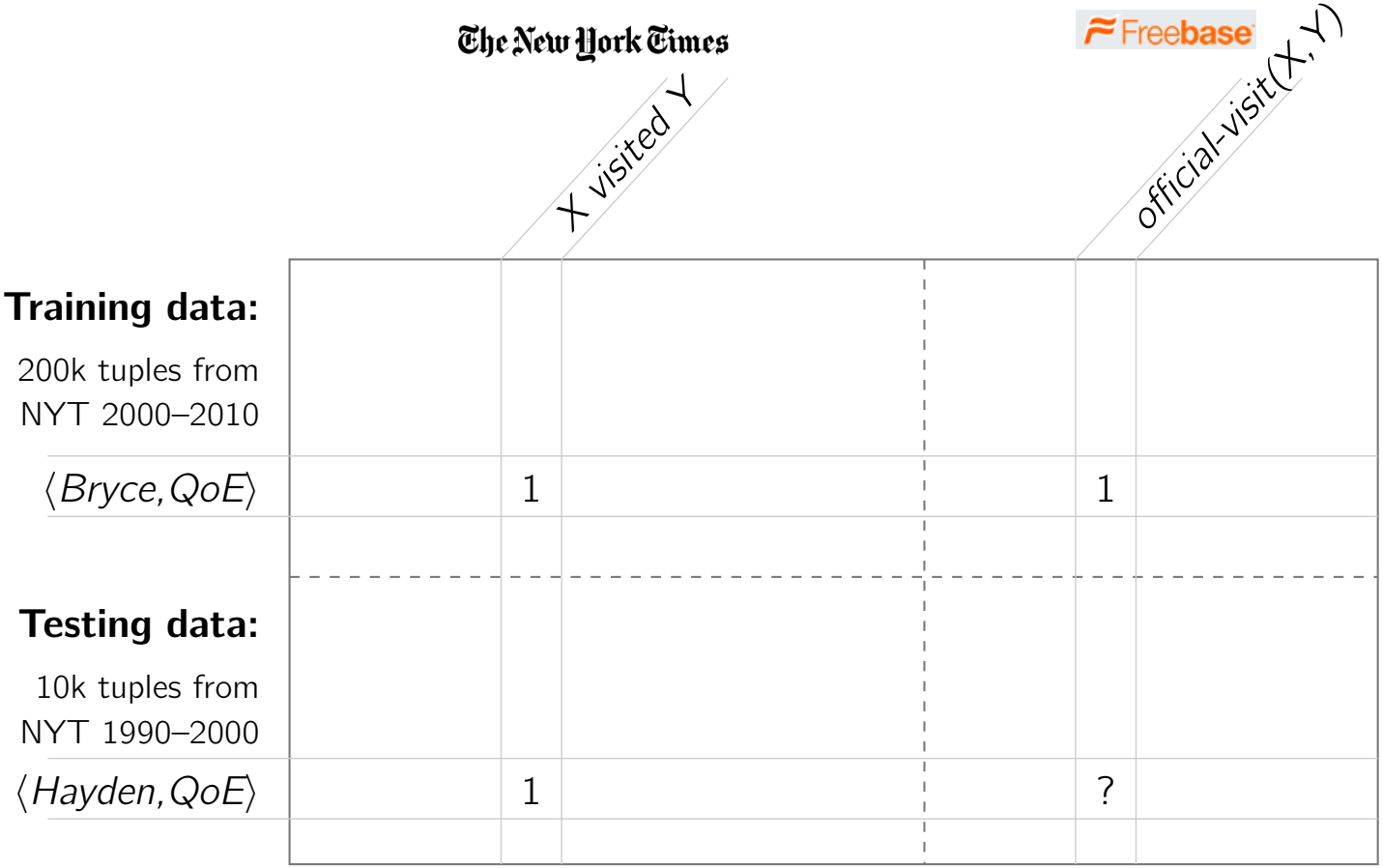
- The future is like the past, i.e., choose the completion which is as close as possible to the known events
- Binary classifier that predicts how likely the future events are given the past events
  - project events onto a finite set of event types
- *n*-gram and Hidden Markov Models
  - linearize events into a sequence
- Hierarchical models of Life Stories
  - a Life Story is a (possibly overlapping) sequence of *careers*
  - each *career* is a sequence of *events*
  - each *event* has *properties* and a *duration*

Outline

FreeBase "notable types" as Life Story summaries

- Life Stories intuition: *background information about entities can improve language understanding*
  - goal: demonstrate that FreeBase's entity information improves an NLP application
- FreeBase entities are assigned *notable types*
  - examples: politician, celebrity, academic, cricket player, etc.
- Methodology:
  - take a state-of-the-art *relation extraction system*
    - Riedel, Yao, McCallum and Marlin (2013) *Relation Extraction with Matrix Factorization and Universal Schemas*
  - modify it to exploit notable types
  - measure the change in relation extraction accuracy

Relation extraction as matrix completion



- *Rows* encode *entity mid tuples* linked from New York Times text
  - named entities in text linked to FreeBase mids with Stanford NEL
- *Columns* encode *relations* in the text or in FreeBase

Reidel's RE model

- $\theta_{r,t}$ is the *score* of the combination of *relation* r appearing with *entity tuple* t

$$\begin{aligned} \mathsf{P}_{r,t}(V{=}1) &= \sigma(\theta_{r,t}), \text{ where:} \\ \sigma(x) &= \frac{1}{1+e^{-x}} \quad \text{is the } \textit{sigmoid function} \end{aligned}$$

- The score $\theta_{r,t}$ is computed by summing three submodels:

$$\theta_{r,t} = \theta^{\mathsf{N}}_{r,t} + \theta^{\mathsf{F}}_{r,t} + \theta^{\mathsf{E}}_{r,t}, \text{ where:}$$

  - $\theta^{\mathsf{N}}_{r,t}$ is the score of the *neighbourhood model*
  - $\theta^{\mathsf{F}}_{r,t}$ is the score of the *latent feature model*
  - $\theta^{\mathsf{E}}_{r,t}$ is the score of the *entity model*

- We modify each of these submodels to exploit notable types

$\Rightarrow$ Improves each submodel, as well as the combined model (?)

The neighbourhood model $\theta^{\mathsf{N}}_{r,t}$

- The neighbourhood model predicts (r, t) entry from other relations r' that t appears with:

$$\theta^{\mathsf{N}}_{r,t} = \sum_{\langle r',t\rangle \in \mathcal{O}\setminus\{\langle r,t\rangle\}} w_{r,r'}$$

  where $\mathcal{O}$ is *set of (relation,entities) tuples in training data*
- Equivalent to *logistic regression classifier* predicting along rows
  - e.g., predict *official-visit(X,Y)* from *X-visited-Y* and ...
  - columns are *features of logistic regression classifier*
- Notable type extension: add columns encoding notable types
  - E.g., add columns *politician(X)* and *politician(Y)*
  - Intuition: politicians are more likely to have official visits
- Increases MAP (mean average precision) from 0.23 to 0.24 but decreases weighted MAP from 0.37 to 0.36 (!)

The latent feature model $\theta^{\mathsf{F}}_{r,t}$

- Each relation r and entity tuple t is associated with 100-dimensional *latent feature vectors* $\mathbf{a}_r$ and $\mathbf{v}_t$ respectively:

$$\theta^{\mathsf{F}}_{r,t} = \mathbf{a}_r^\top \mathbf{v}_t$$

  - the latent feature vectors are *learned from the data* (dimensionality reduction)
- Notable type extension: associate each *notable type pair* b with 100-dimensional latent feature vector v'_b.

$$\theta^{\mathsf{F}'}_{r,t} = \mathbf{a}_r^\top (\mathbf{v}_t + \mathbf{v}'_{b(t)})$$

  where $b(t)$ is *notable type pair* associated with entity tuple t
- Intuition: entity tuples with same types have similiar features
  - L2 regularisation prefers "small" $\mathbf{v}_t$
- Increases MAP from 0.35 to 0.42, and weighted MAP from 0.36 to 0.57

The bilinear entity model $\theta^{\mathsf{E}}_{r,t}$

- Riedel et al's original entity model has low predictive power (?)

$\Rightarrow$ We propose a new *bilinear entity model*

$$\theta^{\mathsf{E}}_{r,t} \;=\; \mathbf{u}_{t_1}^{\top}\, \mathbf{D}_r\, \mathbf{u}_{t_2}$$

where $\mathbf{u}_e$ is a 100-dimensional feature vector for entity e and $\mathbf{D}_r$ is a 100×100-dimensional feature matrix for relation r

- Notable type extension: Associate each notable type z with 50-dimensional latent feature vector $\mathbf{u}'_z$

$$\theta^{\mathsf{E}'}_{r,t} \;=\; \left(\mathbf{u}_{t_1} + \mathbf{u}'_{z(t_1)}\right)^{\top} \mathbf{D}_r \left(\mathbf{u}_{t_2} + \mathbf{u}'_{z(t_2)}\right)$$

where $z(e)$ is the notable type of entity e

- Increases MAP from 0.37 to 0.40, and weighted MAP from 0.43 to 0.46

The combined model $\theta_{r,t} = \theta^{\mathbf{N}}_{r,t} + \theta^{\mathbf{F}}_{r,t} + \theta^{\mathbf{E}}_{r,t}$

Relation	#	MI09	YA11	SU12	N	N^t	F	F^t	E	B	B^t	NF	NF^t	$(NF)^t$	NFE	NFB	$N(FB)^t$	$(NFB)^t$
person/company	146	0.59	0.64	0.64	0.68	0.70	0.76	0.82	0.69	0.61	0.72	0.77	0.85	**0.85**	0.77	0.76	0.81	0.84
location/containedby	89	0.46	0.48	0.51	0.41	0.38	0.66	0.64	0.39	0.32	0.40	0.66	0.64	0.65	**0.67**	0.66	0.60	0.62
person/nationality	55	0.07	0.22	0.06	0.09	0.06	0.11	**0.55**	0.24	0.15	0.32	0.10	0.45	0.39	0.14	0.13	0.43	0.43
person/parents	44	0.04	0.04	0.09	0.12	0.12	0.10	0.35	0.18	0.15	0.30	0.11	**0.36**	0.31	0.10	0.11	0.27	0.29
author/works_written	40	0.36	0.38	0.38	0.34	0.29	0.47	0.52	0.46	0.33	0.47	0.48	0.52	0.52	**0.55**	0.46	0.49	0.51
person/place_of_birth	37	0.38	0.36	0.40	0.26	0.24	0.44	0.42	0.28	0.26	0.23	0.42	0.40	0.39	0.46	0.42	0.39	**0.46**
parent/child	36	0.07	0.13	0.32	0.26	0.25	0.40	0.59	0.39	0.38	0.37	0.42	**0.59**	0.56	0.43	0.46	0.48	0.52
person/place_of_death	26	0.58	0.58	0.63	0.66	0.48	0.63	0.65	0.66	0.67	**0.68**	0.64	0.65	0.63	0.64	0.60	0.64	0.64
neighborhood/neighborhood_of	20	0.00	0.00	0.05	0.27	0.27	0.40	0.46	0.44	0.45	0.45	0.41	0.46	0.46	0.46	0.43	**0.55**	0.51
roadcast/area_served	11	0.18	0.09	0.18	0.11	0.11	0.11	0.20	0.27	0.21	0.26	0.15	0.20	0.22	0.19	0.12	0.25	**0.31**
team_owner/teams_owned	10	0.00	0.10	0.14	0.12	0.13	0.08	0.15	0.33	0.33	**0.39**	0.13	0.15	0.14	0.16	0.14	0.16	0.15
company/founders	9	0.11	0.11	0.23	0.12	0.14	0.35	0.29	0.20	0.20	0.26	**0.36**	0.29	0.27	0.31	0.24	0.27	0.25
team/arena_stadium	9	0.00	0.02	0.02	0.02	0.04	0.03	0.05	**0.12**	0.10	0.10	0.04	0.04	0.04	0.03	0.04	0.03	0.05
film/directed_by	6	0.04	0.10	0.17	0.06	0.05	0.17	0.07	0.27	0.17	0.23	0.18	0.10	0.19	0.21	0.19	**0.29**	0.29
composer/compositions	6	0.00	0.00	0.00	0.17	0.18	0.22	0.35	0.16	0.20	**0.37**	0.28	0.29	0.12	0.05	0.18	0.11	0.13
sports_team/league	6	0.00	0.29	0.12	0.14	0.25	0.39	0.41	0.48	**0.65**	0.59	0.47	0.37	0.37	0.42	0.31	0.36	0.41
person/religion	6	0.00	0.17	0.17	0.09	0.05	0.17	0.18	0.29	0.17	**0.35**	0.17	0.18	0.18	0.17	0.17	0.07	0.13
film/produced_by	4	0.25	0.25	0.25	0.26	0.27	0.13	0.27	0.21	0.18	0.21	0.13	**0.28**	0.07	0.09	0.13	0.21	0.16
structure/architect	2	0.00	0.00	*1.00*	0.27	0.58	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
MAP		0.16	0.21	0.28	0.23	0.24	0.35	**0.42**	0.37	0.34	0.40	0.36	0.41	0.39	0.36	0.34	0.39	0.40
Weighted MAP		0.32	0.37	0.38	0.37	0.36	0.48	**0.57**	0.43	0.38	0.46	0.49	**0.57**	0.56	0.50	0.48	0.53	0.56

N: neighbourhood model, F: latent feature model,
E: entity model, B: bilinear entity model,
X^T: model(s) with notable types

MACQUARIE UNIVERSITY

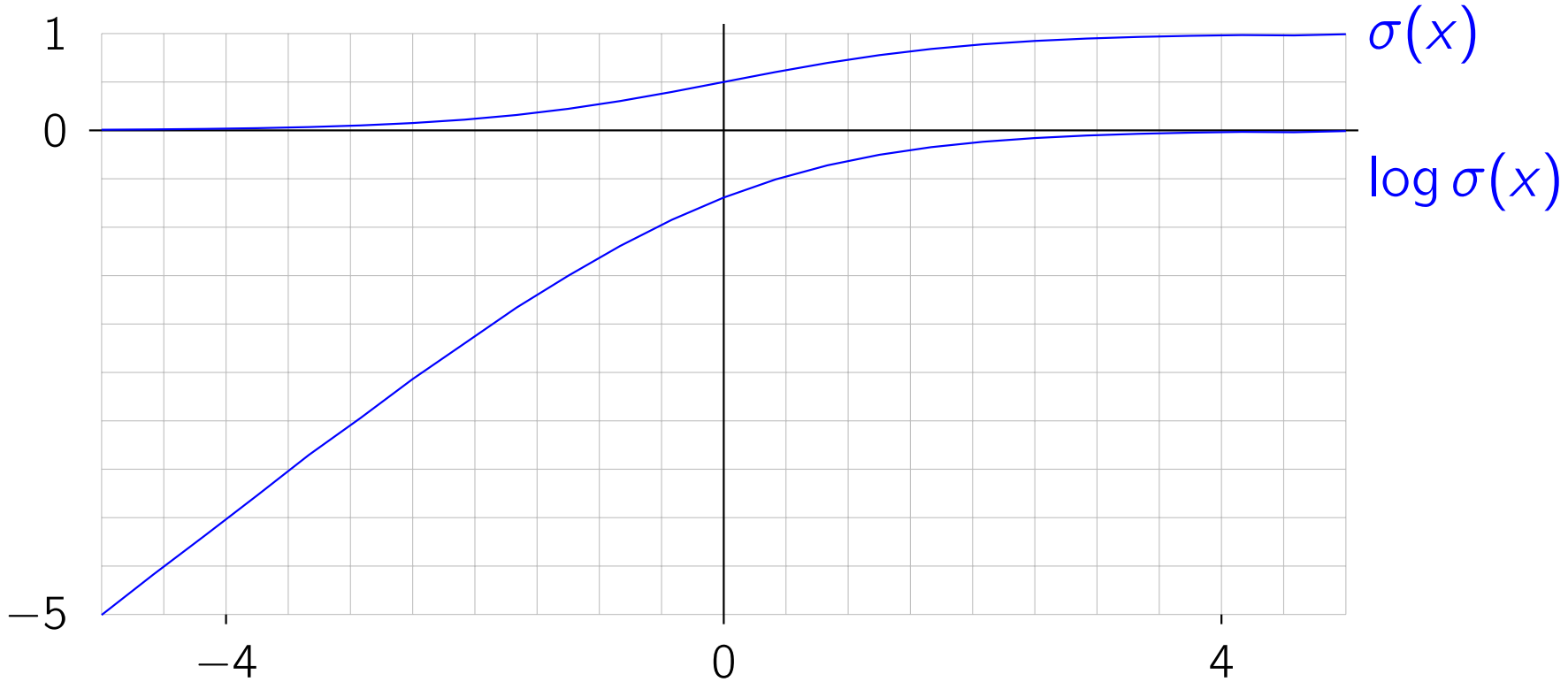
Outline

Parameter estimation as optimisation

- Search for parameters that optimise *objective function* Obj

$$\mathsf{Obj} = \sum_{\langle r,t^+\rangle \in \mathcal{O}} \sum_{\langle r,t^-\rangle \notin \mathcal{O}} \mathsf{Obj}_{\langle r,t^+\rangle,\langle r,t^-\rangle}, \text{ where:}$$

$$\mathsf{Obj}_{\langle r,t^+\rangle,\langle r,t^-\rangle} = \log\left(\sigma(\theta_{\langle r,t^+\rangle} - \theta_{\langle r,t^-\rangle})\right)$$



Stochastic gradient ascent

- Goal: given a multivariate function f, find $\operatorname{argmax}_{\mathbf{x}} f(\mathbf{x})$
- *Gradient ascent:*

  initialise $\mathbf{x}^{(0)}$ somehow (e.g., randomly)

  for $t = 0, 1, 2, \ldots$:

  $$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \epsilon \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}^{(t)})$$

- *Stochastic gradient ascent* uses a stochastic approximation to the gradient:
  - If $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x})$ and i is chosen randomly from $1, \ldots, n$, then:

$$\frac{\partial f}{\partial \mathbf{x}} = n \, \mathsf{E}\left[\frac{\partial f_i}{\partial \mathbf{x}}\right]$$

- Used with the $\log \sigma$ objective function, this yields a *continuous approximation to an error-driven weight update rule*

Outline

Conclusion and future work

- *Machine Learning* (ML) is generally focused on *prediction*
- Learning is usually *reduced to optimisation*
  - very large data sets and numbers of predictors

  $\Rightarrow$ can only approximately optimise
- ML methods apply to *Natural Language Processing*:
  - syntactic parsing
  - entity linking
  - relation extraction
- These methods are being used to construct *large "knowledge databases"*
- Even basic information from databases further improves NLP
  - $\approx$ 10% error reduction by exploiting *notable types*
- "Life Stories" models may let us exploit more information in these databases