# Studio Ousia's Quiz Bowl Question Answering System at NIPS HCQA 2017

Ikuya Yamada[1,2] Ryuji Tamaki[1] Hiroyuki Shindo[3] Yoshiyasu Takefuji[2]

[1]Studio Ousia [2]Keio University [3]Nara Institute of Science and Technology

# Our Background

Tokyo-based tech startup focusing on NLP and deep learning

- QA ENGINE: Deep learning-based question answering system
  - Winner of two competitions
    - **Human-Computer QA @ NIPS 2017**
    - Human-Computer QA @ NAACL 2016
- Semantic Kernel: A state-of-the-art entity linking system
  - Winner of past two competitions:
    - NEEL Challenge @ WWW 2015
    - W-NUT Shared Task #1 @ ACL 2015

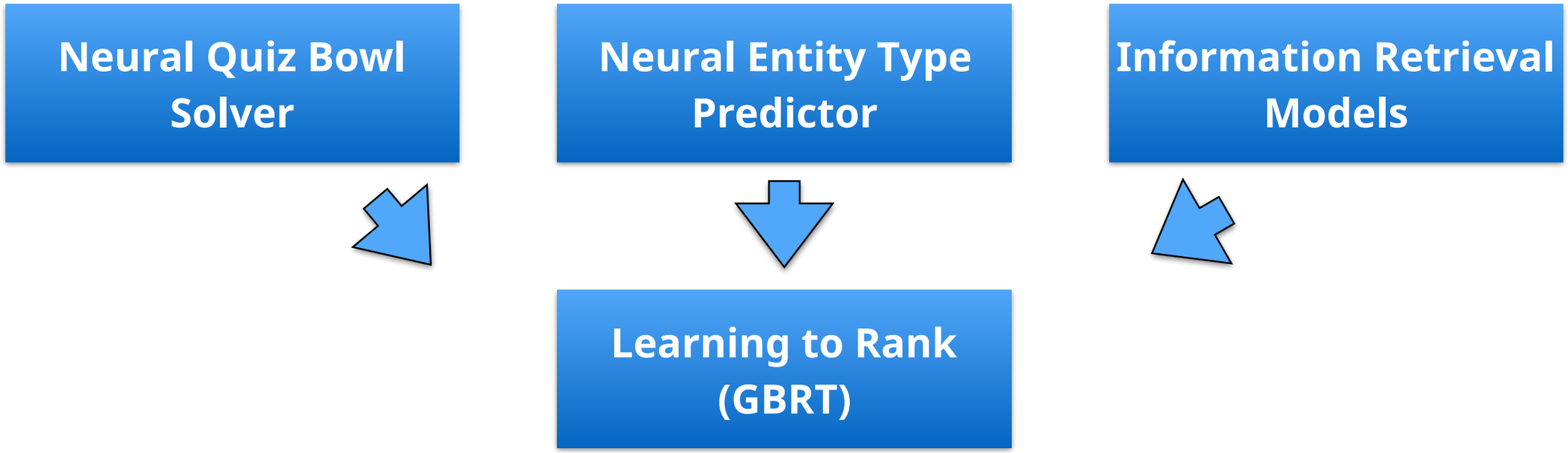
# Summary of the Task

With the assistence of his chief minister, the Duc de Sully, he lowered taxes on peasantry, promoted economic recovery, and instituted a tax on the Paulette. Victor at Ivry and Arquet, he was excluded from succession by the Treaty of Nemours, but won a great victory at Coutras.

Henry IV of France

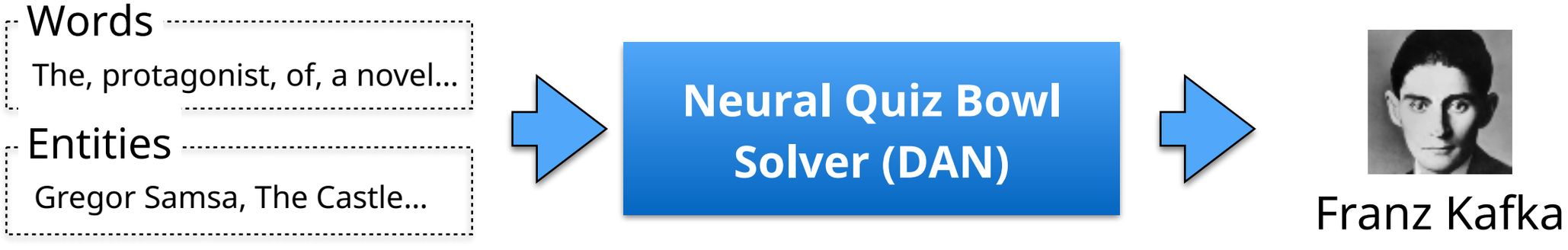
- The system is required to guess the entity that is described in the question
- A question is given one word at a time, and the system can output an answer at any time
- The answer must be an entity that exists in Wikipedia

# Overview of Our Approach




- Two *neural network models* and *information retrieval models* are combined using point-wise learning to rank model
- Our system outputs an answer if the relevance score of a top answer exceeds a threshold

# Neural Quiz Bowl Solver




- *Neural Quiz Bowl Solver* solves the task as a text classification problem over possible answers with words and entities in a question as inputs
- Entities are automatically detected from a question using an entity linking method based on a dictionary built from Wikipedia
- Words, entities, and answers are mapped into continuous vectors which are initialized using Entity-Vector (Yamada et al, 2016)
- The representation of a question is computed based on the average of the word vectors ($\mathbf{a}_w$) and the entity vectors ($\mathbf{b}_e$)

$$\mathbf{v}_{D_w} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{W}_w\mathbf{a}_{w_n},\ \mathbf{v}_{D_e} = \frac{1}{K}\sum_{n=1}^{K}\mathbf{W}_e\mathbf{b}_{e_n}$$

$$\mathbf{v}_D = \mathbf{v}_{D_w} + \mathbf{v}_{D_e}$$

- The probability of each answer being correct is computed based on the dot product between the representation of the question ($\mathbf{v}_D$) and the answer vector ($\mathbf{c}_e$):

$$P(e_t|D) = \frac{\exp(\mathbf{c}_{e_t}{}^{\top}\mathbf{v}_D)}{\sum_{e'\in\Gamma}\exp(\mathbf{c}_{e'}{}^{\top}\mathbf{v}_D)}$$

# Neural Entity Type Predictor


- *Neural Entity Type Predictor* predicts the entity types of the answer (e.g., *person*, *author*) given a question
- Model:  multi-class text classification using convolutional neural networks (CNN) (Kim, 2014)
- Target labels: entity types (automatically assigned to each answer using FIGER entity type set (Ling and Weld, 2012))

# Training of Neural Network Models

- Dataset: 101,043 question-answer pairs created based on the dataset provided by competition organizers

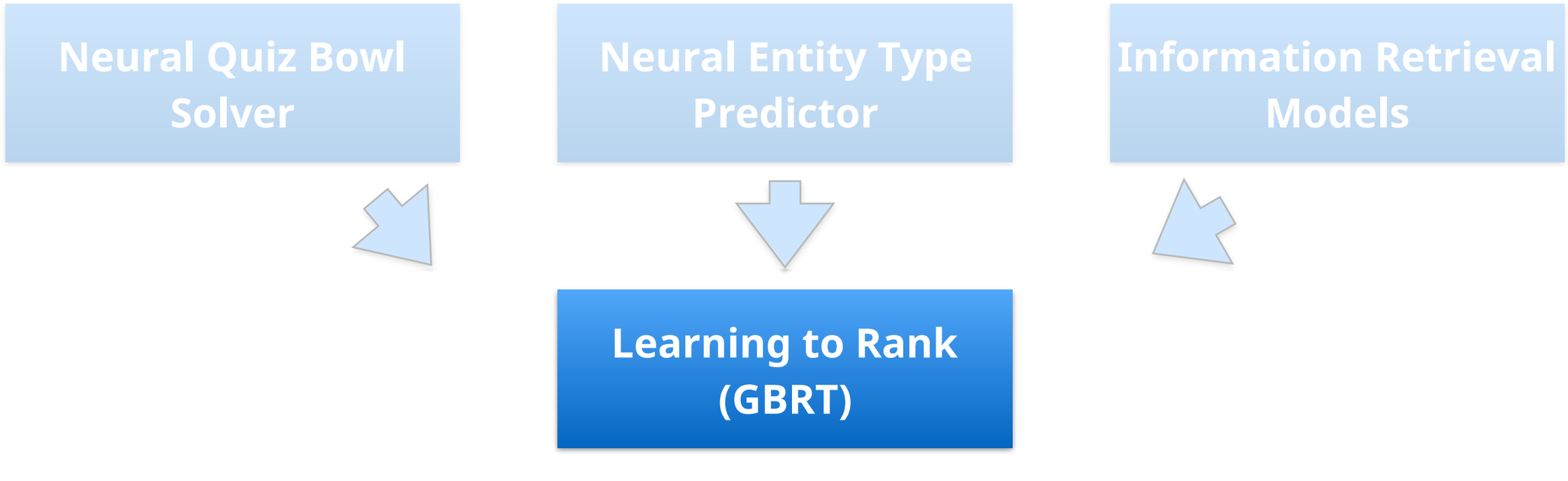
- To perform accurately for incomplete questions, a question is truncated at a random position before inputting it to the models during training

- Stochastic gradient descent (SGD) on GPU is used to train the models

# Information Retrieval Models

- Multiple relevance scores are computed against the documents associated to the answer
- Query: words in a question
- Target documents:
  - *Wikipedia text*: Page text in the answer's Wikipedia entry
  - *Questions in the dataset*: Questions associated to the answer in our dataset
- Scoring methods:
  - Okapi BM25
  - The number of common words between the question and document

# Supervised Point-wise Learning to Rank




- The learning to rank model assigns a relevance score to each answer
- Algorithm: *Gradient boosted regression trees (GBRT)*
- Features: the outputs of the neural network models and the IR models
- The system outputs an answer if the relevance score of a top answer exceeds a threshold, which is set as 0.6
- For each question, we generate five questions truncated at random positions to maintain the accuracy for incomplete questions

# THANK YOU!