**Research Question:** Recent machine learning algorithms are very successful at generating text outputs from text inputs. Can we generate text outputs from database facts instead?

**Key Challenge:** Some output words (e.g., “Basel”) may not have been observed in training data. Regular models cannot be applied on demand to generate new words that have never been seen before.

**Solution:** For on-demand generation of new words, we design a new deep neural architecture that can flexibly incorporate arbitrary items from the fact representations into the output.

---

### Goal

Swiss Tennis Player born in Basel

---

### Model

**Facts**

- Spouse: Mirka Federer
- Height: 185 cm
- Country of Citizenship: Switzerland
- Date of Birth: 8 August 1981
- Place of Birth: Basel
- Occupation: Tennis Player

**Fact Encoder**

- **Input:** set of N facts \( \{f_1, f_2, \ldots, f_N\} \)
- **Output:** concatenation of Fact Embeddings \( \{f_1, f_2, \ldots, f_N\} \)
- **Learn Fact Embeddings using Word Embeddings**
- **Positional Encoder:**
  \[
  f_i = \sum_{j=1}^{N} w_j \cdot \omega(i) \cdot \delta(j)
  \]
- **Mean Fact:** \( f_{N+1} = \frac{1}{N} \sum_{i=1}^{N} f_i \)

**Sequence Decoder**

- **We use Gated Recurrent Unit (GRU) as a sequence decoder**
- **Fact selection:** At each timestep \( t \), the decoder chooses a fact from the set of \( N + 1 \) facts
  \[
  e_t = W_2 \tanh(W_1 [f_t, h_{t-1}]) \\
  f_t = \arg \max_{i \in \{1, N + 1\}} P(f_t | f_i, h_{t-1}) \\
  P(f_t | f_i, h_{t-1}) = \frac{\exp(e_t)}{\sum_{j \in \{1, N + 1\}} \exp(e_j)} \\
  P(f_t | f_i, h_{t-1})
  \]
- **Generating vocabulary words:** If the selected fact is the Mean Fact, the model generates a non-factual vocabulary word
  \[
  o_t = W_e \max(W_e [c_t, h_{t-1}]) \\
  P(w_t | o_t, h_t) = \text{Softmax}(o_t) \\
  w_t = \arg \max_{w \in \mathbb{V}} P(w_t | o_t, h_t)
  \]
- **Copying factual words:** If a fact other than the Mean Fact is selected, the decoder copies a word from the corresponding fact to the output sequence. The position of the word to copy is determined as follows:
  \[
  r_t = W_e \max(W_e [c_t, h_t]) \\
  P(n_t | f_t, h_t) = \text{Softmax}(r_t) \\
  n_t = \arg \max_{n \in \{1, \ldots, |f_t|\}} P(n_t | f_t, h_t)
  \]

---

### Results

Examples of generated descriptions for Wikidata entities with missing descriptions

---

### Code and Data

https://github.com/kingsaint/Wikidata-Descriptions