A Shared Task on Bandit Learning for Machine Translation

Artem Sokolov⋄, Julia Kreutzer*, Kellen Sunderland⋄, Pavel Danchenko⋄,
Witold Szymaniak⋄, Hagen Fürstenau⋄, Stefan Riezler*,†

amazon  Universitätsrechenzentrum Heidelberg

WMT17
<table>
<thead>
<tr>
<th>MT supervision</th>
<th>cost</th>
<th>noise/signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>references</td>
<td>$$$</td>
<td>low</td>
</tr>
<tr>
<td>post-edits/quasi.refs</td>
<td>$</td>
<td>high</td>
</tr>
<tr>
<td>feedback</td>
<td>$</td>
<td>high</td>
</tr>
</tbody>
</table>

Motivation

Sweet spot for bandit MT:
1. Costs drop faster than noise/signal increases ⇒ lots of data
2. Downstream performance is hard to wrap into an automatic metric
## Motivation

### References

<table>
<thead>
<tr>
<th>MT supervision</th>
<th>cost</th>
<th>noise/signal</th>
<th>downstream fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>references</td>
<td>$$$</td>
<td>low</td>
<td>hard</td>
</tr>
<tr>
<td>post-edits/ quasi.refs</td>
<td>$</td>
<td>middle</td>
<td>hard</td>
</tr>
<tr>
<td>feedback</td>
<td>$</td>
<td>high</td>
<td>easy/trivial</td>
</tr>
<tr>
<td>MT supervision</td>
<td>cost</td>
<td>noise/signal</td>
<td>downstream fit</td>
</tr>
<tr>
<td>----------------</td>
<td>------</td>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>references</td>
<td>$$$</td>
<td>low</td>
<td>hard</td>
</tr>
<tr>
<td>post-edits/quasi.refs</td>
<td>$$</td>
<td>middle</td>
<td>hard</td>
</tr>
<tr>
<td>feedback</td>
<td>$</td>
<td>high</td>
<td>easy/trivial</td>
</tr>
</tbody>
</table>

**Sweet spot for bandit MT:**

1. costs drop faster than noise/signal increases \(\Rightarrow\) lots of data
2. downstream performance is hard to wrap into an automatic metric
One-armed bandits – slot-machines:
- pull an arm to play
- get some reward (or none)
- try a new machine or stick to the best among discovered so far?
Multi-armed bandits:
- many arms (actions)
- each arm has an unknown reward distribution
- find an arm-picking strategy to maximize total reward
Multi-armed bandits for structured prediction (MT):

- observe context (source sentence)
- pick one out of exponentially many outputs (translations)
- each structure results in some reward (BLEU)
- tune an arm-picking strategy (decoder weights)
Multi-armed bandits for structured prediction (MT):

- observe context (source sentence)
- pick one out of exponentially many outputs (translations)
- each structure results in some reward (BLEU)
- tune an arm-picking strategy (decoder weights)
Multi-armed bandits for structured prediction (MT):

- observe context (source sentence)
- pick one out of exponentially many outputs (translations)
- each structure results in some reward (BLEU)
- tune an arm-picking strategy (decoder weights)
- note: only one translation is scored, others are not
for $t = 0, \ldots, T$ do
    Request source sentence $x_t$ from service
    Propose a translation $y_t$
    Obtain feedback $\Delta(y_t)$ from service
    Improve MT model
- DE-EN (pre-processed)
- domain-adaptation: general (WMT17) $\rightarrow$ e-commerce (Amazon)
- all participants received the same sequence of $x_t$
- feedback was sent-BLEU (plans for more realistic feedback dropped to avoid complicating the task)
- organizers provided the service, client SDK and baselines
Data & Phases

- DE-EN (pre-processed)
- domain-adaptation: general (WMT17) → e-commerce (Amazon)
- all participants received the same sequence of $x_t$
- feedback was sent-BLEU (plans for more realistic feedback dropped to avoid complicating the task)
- organizers provided the service, client SDK and baselines

Participants had to do:

1. pick Python or Java
2. download a short client snippet
3. wrap it around an MT system
- DE-EN (pre-processed)
- domain-adaptation: general (WMT17) → e-commerce (Amazon)
- all participants received the same sequence of $x_t$
- feedback was sent-BLEU (plans for more realistic feedback dropped to avoid complicating the task)
- organizers provided the service, client SDK and baselines

**Participants had to do:**
1. pick Python or Java
2. download a short client snippet
3. wrap it around an MT system

<table>
<thead>
<tr>
<th>phase</th>
<th>sentences</th>
<th>passes</th>
<th>purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>mock (since 13 Mar)</td>
<td>40</td>
<td>unlimited</td>
<td>test client API</td>
</tr>
<tr>
<td>dev (since 5 Apr)</td>
<td>40k</td>
<td>unlimited</td>
<td>tune hyperparams</td>
</tr>
<tr>
<td><strong>train</strong> (25 Apr - Jun 9)</td>
<td>1.3M</td>
<td>only one</td>
<td>final evaluation</td>
</tr>
</tbody>
</table>
- **cumulative reward**

\[
\sum_{t=1}^{T} \Delta(y_t)
\]

- **corpus-BLEU** at regular intervals on an embedded test set:
  - 700 sent. at 4 locations in 40k dev sent.
  - 4000 sent. at 12 locations in 1.3M train sent.

- **regret**

\[
\frac{1}{T} \sum_{t=1}^{T} \Delta(y^*_t) - \Delta(y_t)
\]

(average cumulative reward difference w.r.t. to an in-domain system)
Challenges

- different domains \(\Rightarrow\) high OOV rate
- learning from one-shot feedback
- real-world data:
  - typos/errors in sources
  - mixed direction of data
  - translators improved readability/corrected errors/deleted irrelevancies
Challenges

- different domains ⇒ high OOV rate
- learning from one-shot feedback
- real-world data:
  - typos/errors in sources
  - mixed direction of data
  - translators improved readability/corrected errors/deleted irrelevancies

<table>
<thead>
<tr>
<th>source</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>schwarz gr.xxI / xxI</td>
<td>black, size xxI / xxxI</td>
</tr>
<tr>
<td>147 cm</td>
<td>147 cm</td>
</tr>
<tr>
<td>für starke, glänzende nägel</td>
<td>great for strengthen your nails and enhance shine</td>
</tr>
<tr>
<td>seamless verarbeitung</td>
<td>seamless processing</td>
</tr>
<tr>
<td>maschinenwaschbar bei 30 °c</td>
<td>machine washable at 30 degrees.</td>
</tr>
<tr>
<td>32 unzen volumen</td>
<td>32-ounce capacity</td>
</tr>
<tr>
<td>material: 1050 denier nylon</td>
<td>material: 1050d nylon.</td>
</tr>
<tr>
<td>für e-gitarre entworfen</td>
<td>designed for electric guitar</td>
</tr>
</tbody>
</table>
- 8 teams registered
- 4 used the dev service
- 2 started full training:

**Guillaume Wisniewski**

- UCB-style selection from a pool of MT systems
- online linear regression to predict rewards
- additional exploration

**Amr Sharaf, Shi Feng, Khanh Nguyen, Kianté Brantley, Hal Daumé**

- domain-adaptation (Moore-Lewis)
- online non-linear regression for rewards
- policy gradient with adaptive control variate (Advantage Actor-Critic)
- NMT (word- and BPE-based; NeuralMonkey)
- SMT (dense features; cdec)

stochastic approximation of the **expected loss**

**Policy-gradient**

\[
\mathbb{E}_{\tilde{y} \sim p_w(y|x)}[\Delta(\tilde{y})]
\]

**Gradient-free**

\[
\mathbb{E}_{\varepsilon \sim \mathcal{N}(0,1)}[\Delta(\hat{y}(w + \varepsilon))]
\]

For \(t = 1, \ldots, T\) do
- Observe \(x_t\)
- Sample \(\tilde{y}_t \sim p_{w_t}(y|x_t)\)
- Obtain \(\Delta(\tilde{y}_t)\)

\[
w_{t+1} = w_t - \gamma \Delta(\tilde{y}_t) \nabla \log p_{w_t}(\tilde{y}_t|x_t)
\]

For \(t = 1, \ldots, T\) do
- Observe \(x_t\)
- Sample \(\varepsilon_t \sim \mathcal{N}(0, 1)\)
- Decode \(\hat{y}_t\) with \(w_t + \varepsilon_t\)
- Obtain \(\Delta(\hat{y}_t)\)

\[
w_{t+1} = w_t + \gamma \Delta(\hat{y}_t) \varepsilon_t
\]
Checkpoints (sent-BLEU)

![Checkpoint Graph](image)

- SMT-static
- SMT-SZO-CV-ADAM
- SMT-EL-CV-ADAM
- BNMT-static
- WNMT-EL
- BNMT-EL
- UMD-dom-adapt
Static system:

- UMD domain adaptation got the best BLEU and lowest regret
Results

Static system:

⭐ UMD domain adaptation got the best BLEU and lowest regret

Learning systems:

- cumulative reward:
  ⭐ BNMT-EL is the only to beat its static NMT baseline
    - lowest regret of all learning systems
    - best sent-BLEU overall
  ⭐ SMT-EL-CV came very close

- corpus-BLEU:
  ⭐ SMT-EL-CV improves over its SMT baseline
  ➡️ none of the submissions show monotonic learning curves (or are too short)

Conclusion:

I difficult task (even with all simplifications)
I difficult data
I we need more research!:)

14 / 14
**Static system:**

- UMD domain adaptation got the best BLEU and lowest regret

**Learning systems:**

- Cumulative reward:
  - BNMT-EL is the only to beat its static NMT baseline
    - +lowest regret of all learning systems
    - +best sent-BLEU overall
  - SMT-EL-CV came very close

- Corpus-BLEU:
  - SMT-EL-CV improves over its SMT baseline
    - none of the submissions show monotonic learning curves
    - (or are too short)

**Conclusion:**

- Difficult task (even with all simplifications)
- Difficult data
- We need more research!:)
Acknowledgments

Funding

Amazon

DFG

Task idea, NMT baselines

API, SDK, leaderboard & operation

task design, data, SMT baselines, etc.
Acknowledgments

funding

Amazon

DFG

task idea, NMT baselines and bandit pictures!

API, SDK, leaderboard & operation

task design, data, SMT baselines, etc.

Thanks!