Hyper-HaBERTor: a light-weight pretrained hatespeech language detector using hypercomplex space

Thanh Tran, Kyumin Lee
Department of Computer Science and Data Science
Worcester Polytechnic Institute
MA, USA
2020
What is hatespeech?

- **Hatespeech**: refer to the speech that **conveys hateful or discriminatory or stereotyped ideas** on the basis of factors such as: **race, gender, religion, sexual orientation**.
Why detecting hatespeech?

Easy to spread hatespeech to a large number of online users.
Why detecting hatespeech?

Hatespeech hurts targeting people:

- **Restrict** their **freedom of speech** on social medias.
- **Interfere** with **civil discourse**.
- **Turn good people away** from saying out loud their opinions.
What’s wrong with existing hatespeech detectors?

- Some of the first works design features manually and then feed into traditional machine learning models (i.e. SVM, Logistic Regression):
  - → Limited by the quality and quantity of the human-crafted features.

- Recent works used deep neural network:
  - CNN based models: [1], [2].
  - RNN based models: [3]
  - → Those models lack the ability of Language Understanding and the results are sensitive with weight initialization methods.

What’s wrong with existing hatespeech detectors?

• Recent pretrained language model (BERT-base [1]) has shown its superior in NLP tasks by pretraining a language model on a vast amount of texts.
  
  — While hatespeech texts have unique properties compared to normal texts (i.e. using a lot of asterisks like: f*ck, ...; or letters are often replaced by numbers: i → 1, g → 9, ...)
  
  — → Language model like BERT-base model is still limited in two points:
  
  ▪ Lack of hatespeech language understanding because they are pretrain on non-hatespeech (formal) corpus like Wiki, Book corpus.
  
  ▪ Has a lot number of parameters, which cause a lot of (GPU) memory to train.

• Recent efforts aim to reduce BERT-base complexity by knowledge distillation method,
  
  — Still lack of hatespeech language understanding, and performed worse than BERT-base.

Motivation

Can we build a pretrained Language Model that

• Has a **smaller number of parameters?**
  – **Solution:** Using Quaternion representations and Quaternion feed forward transformation.

• With a **better or equivalent** performance on hatespeech detection task?
  – **Solution:** Pretrain from scratch a language model on hatespeech related corpus to equip the model with some hatespeech language understanding.
What is Quaternion representations?

- Quaternion number:
  \[ X = r + a \, i + b \, j + c \, k \]
  \[(ijk = i^2 = j^2 = k^2 = -1; \, ij = k; \, jk = i; \, ki = j; \, ji = -k; \, kj = -i; \, ik = -j)\]
  For example: \[ X = 5 + 2i + 3j + 4k \]

- Hamilton product between two Quaternions X and Y (\( X \otimes Y \)):
  \[ X \otimes Y = r_X \, r_Y - a_X \, a_Y - b_X \, b_Y - c_X \, c_Y + \]
  \[(r_X a_Y + a_X r_Y + b_X c_Y - c_X b_Y) \, i + \]
  \[(r_X b_Y - a_X c_Y + b_X r_Y + c_X a_Y) \, j + \]
  \[(r_X c_Y + a_X b_Y - b_X a_Y + c_X r_Y) \, k \]

\[
X \otimes Y = [r_X, a_X, b_X, c_X]
\begin{bmatrix}
  r_Y & a_Y & b_Y & c_Y \\
  -a_Y & r_Y & -c_Y & b_Y \\
  -b_Y & c_Y & r_Y & -a_Y \\
  -c_Y & -b_Y & a_Y & r_Y
\end{bmatrix}
\]
Why Quaternion representation?

- **Benefits:**
  - Provide a **better inter-dependencies interaction coding** due to the weight sharing in Hamilton product.
  - **Reduce 75% of the number of parameters** over real-valued representations in Euclidean space.
  - Recent works have shown its great performance in NLP and computer vision [1,2,3].

Our model: Hyper-HaBERTor = Quaternion + BERT + pretrain on hatespeech corpus.

- Architecture:
Comparision between BERT-base and Hyper-HaBERTor

### How many parameters can we reduce?

<table>
<thead>
<tr>
<th>Component</th>
<th>From</th>
<th>To</th>
<th>#params</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Vocab</strong></td>
<td></td>
<td></td>
<td>24,576,000</td>
</tr>
<tr>
<td>1 encoder layer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Self-attention Transform key</td>
<td>768</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td>Transform query</td>
<td>768</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td>Transform value</td>
<td>768</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td>output</td>
<td>768</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td>Intermediate</td>
<td>768</td>
<td>3,072</td>
<td>2,359,296</td>
</tr>
<tr>
<td>Output</td>
<td>3,072</td>
<td>768</td>
<td>2,359,296</td>
</tr>
<tr>
<td>1 encoder layer</td>
<td></td>
<td></td>
<td><strong>7,077,888</strong></td>
</tr>
<tr>
<td>12 encoder layers</td>
<td></td>
<td></td>
<td>84,934,656</td>
</tr>
<tr>
<td>Pooling</td>
<td>768</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>110,100,480</strong></td>
</tr>
</tbody>
</table>

**BERT-base**

<table>
<thead>
<tr>
<th>Component</th>
<th>From</th>
<th>To</th>
<th>#params</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Vocab</strong></td>
<td></td>
<td></td>
<td>24,576,000</td>
</tr>
<tr>
<td>1 encoder layer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Self-attention Transform key</td>
<td>768</td>
<td>768</td>
<td>147,456</td>
</tr>
<tr>
<td>Transform query</td>
<td>768</td>
<td>768</td>
<td>147,456</td>
</tr>
<tr>
<td>Transform value</td>
<td>768</td>
<td>768</td>
<td>147,456</td>
</tr>
<tr>
<td>output</td>
<td>768</td>
<td>768</td>
<td>147,456</td>
</tr>
<tr>
<td>Intermediate</td>
<td>768</td>
<td>3,072</td>
<td>589,824</td>
</tr>
<tr>
<td>Output</td>
<td>3,072</td>
<td>768</td>
<td>589,824</td>
</tr>
<tr>
<td>1 encoder layer</td>
<td></td>
<td></td>
<td><strong>1,769,472</strong></td>
</tr>
<tr>
<td>12 encoder layers</td>
<td></td>
<td></td>
<td>21,233,664</td>
</tr>
<tr>
<td>Pooling</td>
<td>768</td>
<td>768</td>
<td>147,456</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>45,957,120</strong></td>
</tr>
</tbody>
</table>

**Hyper-HaBERTor**
Experiments

• Configuration:

• Modification of HyperBert:
  – Using SentencePiece to learn subword vocabs.
  – Real-valued transformation $\rightarrow$ Quaternion transformation.
  – Real-valued embeddings $\rightarrow$ Quaternion embeddings.
  – Multi-head scaled dot attention: concatenate 4 components of Quaternion and do the same logic.
  – 10 training examples for masked token prediction/instance as similar to RoBERTa.
  – Adding targeted adversarial learning for fine-tuning phase.

• Comparison:
  – Baseline: BERT-base (110M), DistillBert (66M), TinyBert-4layers (14.5M), RoBERTa-base (125M, 50k vocab size).
  – Downstream Task: hatespeech detection.
  – Data: Yahoo, Wiki, Twitter.
### Experiments

- **Results on fine-tuning hate-speech task:**

<table>
<thead>
<tr>
<th>Model</th>
<th>Yahoo News</th>
<th></th>
<th></th>
<th></th>
<th>York News</th>
<th></th>
<th></th>
<th></th>
<th>Wiki</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>AP</td>
<td>F1</td>
<td>AUC</td>
<td>AP</td>
<td>F1</td>
<td>AUC</td>
<td>AP</td>
<td>F1</td>
<td>AUC</td>
<td>AP</td>
</tr>
<tr>
<td>TinyBert-4Layers</td>
<td>93.13</td>
<td>71.25</td>
<td>64.69</td>
<td>94.12</td>
<td>60.56</td>
<td>58.01</td>
<td>92.23</td>
<td>83.88</td>
<td>78.33</td>
<td>97.10</td>
<td>87.64</td>
</tr>
<tr>
<td>DistilBert</td>
<td>93.13</td>
<td>71.25</td>
<td>64.69</td>
<td>94.12</td>
<td>60.56</td>
<td>58.01</td>
<td>92.13</td>
<td>80.21</td>
<td>77.89</td>
<td>97.23</td>
<td>88.16</td>
</tr>
<tr>
<td>BERT-base</td>
<td><strong>93.56</strong></td>
<td><strong>71.65</strong></td>
<td><strong>65.30</strong></td>
<td><strong>94.60</strong></td>
<td><strong>62.34</strong></td>
<td><strong>59.72</strong></td>
<td><strong>93.21</strong></td>
<td><strong>86.67</strong></td>
<td><strong>79.68</strong></td>
<td><strong>97.75</strong></td>
<td><strong>89.23</strong></td>
</tr>
<tr>
<td>RoBERTa</td>
<td>92.63</td>
<td>70.15</td>
<td>63.76</td>
<td>93.83</td>
<td>59.73</td>
<td>57.74</td>
<td>90.63</td>
<td>84.36</td>
<td>76.30</td>
<td>95.71</td>
<td>84.48</td>
</tr>
<tr>
<td>HyperBERT - adv</td>
<td>93.55</td>
<td><strong>71.94</strong></td>
<td><strong>65.54</strong></td>
<td>95.20</td>
<td>62.17</td>
<td>59.65</td>
<td>91.88</td>
<td>84.23</td>
<td>78.05</td>
<td>97.06</td>
<td>87.24</td>
</tr>
<tr>
<td>HyperBERT + adv</td>
<td><strong>93.71</strong></td>
<td><strong>72.59</strong></td>
<td><strong>66.08</strong></td>
<td><strong>95.11</strong></td>
<td><strong>62.88</strong></td>
<td><strong>59.88</strong></td>
<td><strong>93.26</strong></td>
<td><strong>86.81</strong></td>
<td><strong>80.21</strong></td>
<td><strong>97.24</strong></td>
<td><strong>88.01</strong></td>
</tr>
</tbody>
</table>

HyperBERT get better results with adversarial learning on fine-tuning downstream task. HyperBERT + adv work best for Yahoo and Twitter, but less than BERT-base in wiki with a small amount. Reason: BERT-base is pretrained on Wiki --> had advantage.
Conclusion

In this talk:

• Utilizing Quaternion space in building a *hatespeech* language model using *Quaternion* representations for the *hatespeech* detection task.
  – First work using *Quaternion representations* on a pretrained language model.
  – First work applies adversarial learning on Quaternion space.

• Hyper-HaBERTor obtained, on average, *slightly better F1 scores* on several hatespeech datasets compared to BERT-base model, while reduce *6 times number* of parameters.
Thank You and Questions?