รายงาน

“DaDa” Student Assistant Chatbot

เสนอ

อ.ดร. พีรพล เวทีกูล

อ.ดร. เอกพล ช่วงสุวนิช

รายชื่อสมาชิกในกลุ่ม

5730174521	ณัฐพล รักษ์รัชตกุล
5730108121	ชนาธิป แซ่เตีย
5730192821	ณัฐสิทธิ์ มหารัตนมาลัย
5730196321	ดนุภัทร คำนวนสินธุ์

โครงการนี้เป็นส่วนหนึ่งของรายวิชา 2110594 (NLP)

ภาคการศึกษาปลาย ปีการศึกษา 2560

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

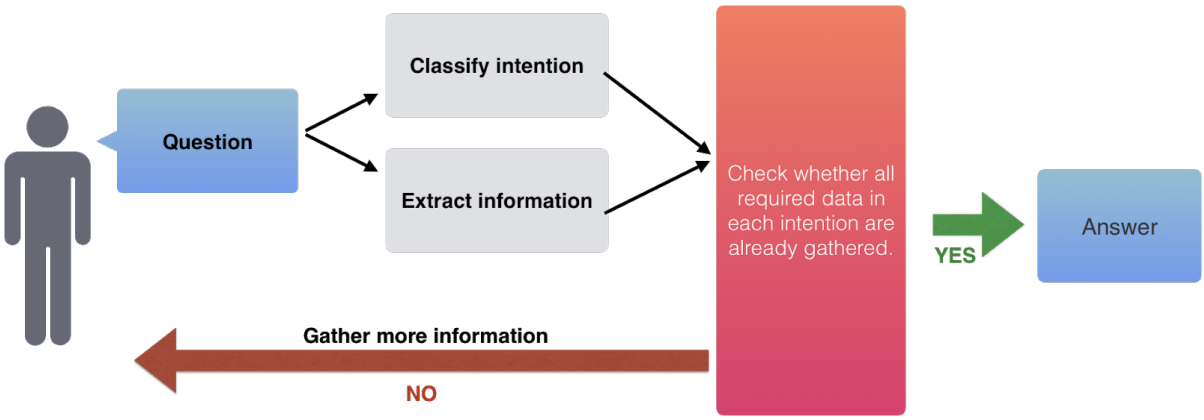
ที่มาของโครงการ

หัวข้อหนึ่งที่นิสิตและนักศึกษาทุกคนมักจะคุยกันก็คงตกไม่พ้นหัวข้อการเรียน และ คำถามทั่วไปของคอร์สเรียนก็มักจะเกิดขึ้นและจะต้องมีผู้ที่ศึกษาข้อมูลมาอย่างดีคอยตอบอยู่บ่อยๆ อีกทั้งคำถามเหล่านั้นยังคงมีอยู่ตลอดปี เช่น เมื่อเราไปเรียนในสถานที่ไม่คุ้นเคยหรือเรียนหลายวิชา หลายครั้งก็ลืมห้องเรียนจนทำให้ต้องเข้าไปหาข้อมูลในเว็บไซต์ซึ่งต้องเข้าหลายขั้นตอนกว่าจะได้คำตอบ หรือหลายคนก็ชอบใช้วิธีถามเพื่อนอยู่บ่อยๆ บางข้อมูลก็เป็นข้อมูลที่ต้องการรู้ทุกครั้งก่อนการเรียนเช่น เนื้อหาวิชาในสัปดาห์นั้น หรือ slide ของเนื้อหาในสัปดาห์นั้น จึงเกิดโครงการ chatbot ผู้ช่วยนิสิตขึ้นมาเพื่อช่วยตอบคำถามที่มันจะถูกถามในหมู่นิสิตและนักศึกษา เพื่อเพิ่มความสะดวกสบายให้แก่นิสิตซึ่งมีจำนวนมาก

วัตถุประสงค์ของโครงการ

- เพื่อช่วยให้นิสิตสอบถามข้อมูลเกี่ยวกับคอร์สเรียนได้รวดเร็วและตลอดเวลา
- เพื่อรวมข้อมูลที่สำคัญและจำเป็นเกี่ยวกับคอร์สเรียนไว้ในที่เดียว
- เพื่อเพิ่มความสะดวกสบายในการแจ้งข้อมูลระหว่างผู้สอนและผู้เรียน
- เพื่อให้นิสิตสามารถทบทวนบทเรียนได้ง่ายยิ่งขึ้น

การทำงานของระบบ



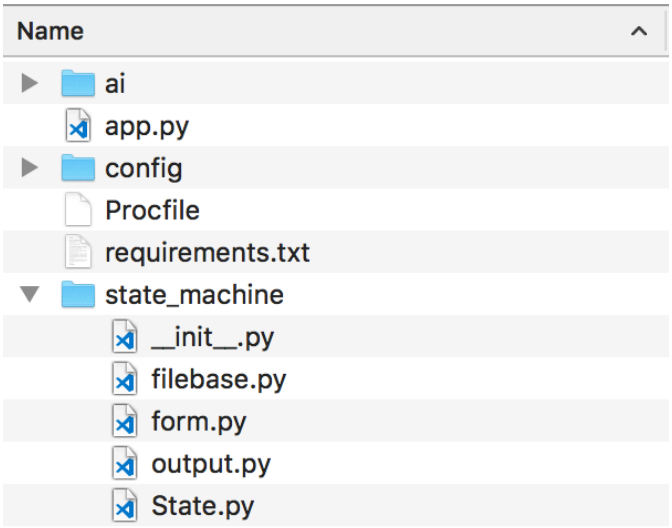
ภาพที่ 1 แผนภาพการทำงานของระบบ Student Assistant Chatbot

เมื่อผู้ใช้ถามคำถามมา ระบบจะทำการแบ่งประเภทของข้อความ รวมถึงดึงข้อมูลที่เกี่ยวข้องต่อการตอบคำถามออกมา หากข้อมูลที่ดึงออกมาเพียงพอต่อการตอบคำถามแล้วระบบจะทำการตอบคำถามกลับไปยังผู้ใช้ แต่หากข้อมูลที่เก็บมายังไม่เพียงพอระบบจะทำการถามคำถามผู้ใช้เพิ่มจนกว่าจะได้ข้อมูลครบถ้วนเพื่อจะตอบคำถาม เมื่อได้ข้อมูลครบถ้วนแล้วก็จะตอบคำถาม

การเชื่อมต่อ

ภาพที่ 2 แผนภาพการเชื่อมต่อของระบบ Student Assistant Chatbot บน heroku

โครงสร้างภายใน Web Application



ภาพที่ 3 โครงสร้างภายในของ Web Application (Flask framework)

- **/ai** จะเป็น package ที่ไว้ใช้สำหรับการทำนาย intention และหา information ต่าง ๆ โดยโค้ตส่วนหลักที่จะทำ data preprocessing และเรียกใช้งาน model จะอยู่ใน ai.py และ ตัว model จะเก็บอยู่ในโฟล์เดอร์ dump
- **app.py** ทำหน้าที่เป็นเหมือนไฟล์ main ของ application คอยติดต่อประสานงานระหว่าง firebase, statemachine และ LINE messaging API
- **/config** นั้นจะเก็บไฟล์ที่เป็นไฟล์ config ต่าง ๆ ดังนี้
    - **require_form.json** - required information ที่ใช้สำหรับการตอบคำถามแต่ละ intention ของ chatbot
    - **answer_require_information.json** - ประโยคที่ใช้ในการตอบคำถามเมื่อบอทต้องการ information แต่ละ information เพิ่ม
- **/state_machine** จะเก็บ package ที่เป็น state machine หลักของ chatbot โดยจะมีไฟล์และการดังนี้
    - **filebase.py** - ฟังก์ชันต่าง ๆ ที่ใช้ในการติดต่อกับ filebase database ต่าง ๆ ทั้งการอ่านและเขียน
    - **form.py** - ฟังก์ชันในการจัดการและตรวจสอบ information ต่าง ๆ ที่ผู้ใช้ได้บอกมา
    - **output.py** - ฟังก์ชันที่ใช้ในการ generate text เพื่อตอบปัญหาต่าง ๆ
    - **State.py** - คลาส state machine หลักที่ chatbot ใช้ในการทำงาน โดยจะรับ input เป็น text จัดการและตรวจสอบ information ต่าง ๆ โดย chatbot จะมีการส่งข้อความในส่วนของ transition ใน state machine

Dataset & Information

เป็นข้อมูลที่ได้จะเป็นข้อความทั้งหมด 400 ข้อความที่ต้องการจะถามข้อมูลต่าง ๆ กับบอท โดยแต่ละข้อความจะมี intention ของแต่ละข้อความกำกับอยู่ โดยจะมี intention ทั้งหมด 7 intention ดังนี้

- แจ้งสถานที่เรียน
- แจ้งเนื้อหาที่เรียน
- ถามเกี่ยวกับตารางเรียน
- ถามเกี่ยวกับอีเวนต์สำคัญ
- ถามสถิติเช็คชื่อ
- ติวก่อนสอบ
- ขอสไลด์เอกสาร

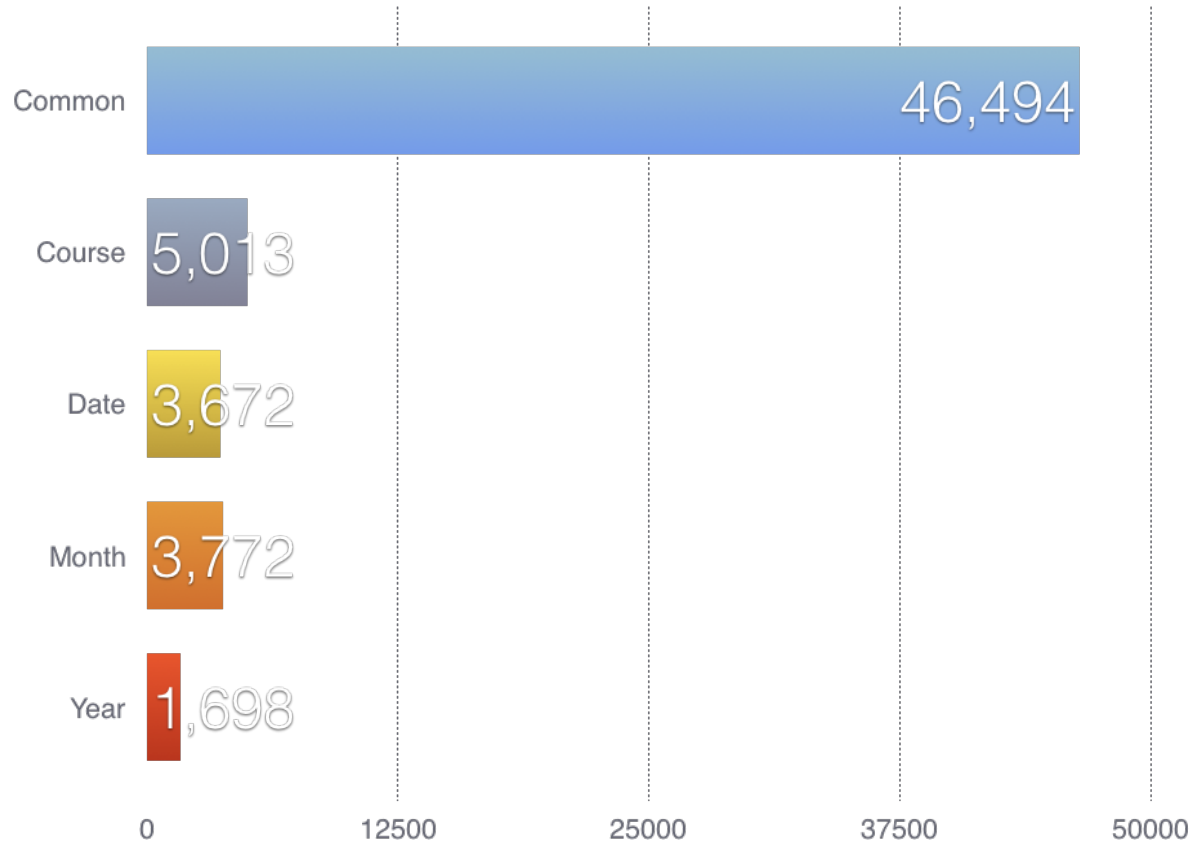
แต่ละข้อความที่ได้นั้นจะมี information ที่จำเป็นต่อการตอบคำถามของบอทในแต่ละ intention โดย information จะมี 5 information ดังนี้

- Common
- Course
- Date
- Month
- Year

ข้อมูลต่าง ๆ ของชุดข้อมูลนี้นั้น สามารถดูได้ใน ตารางและกราฟด้านล่าง

ตารางที่ 1 ข้อมูลเชิงจำนวนของชุดข้อมูล

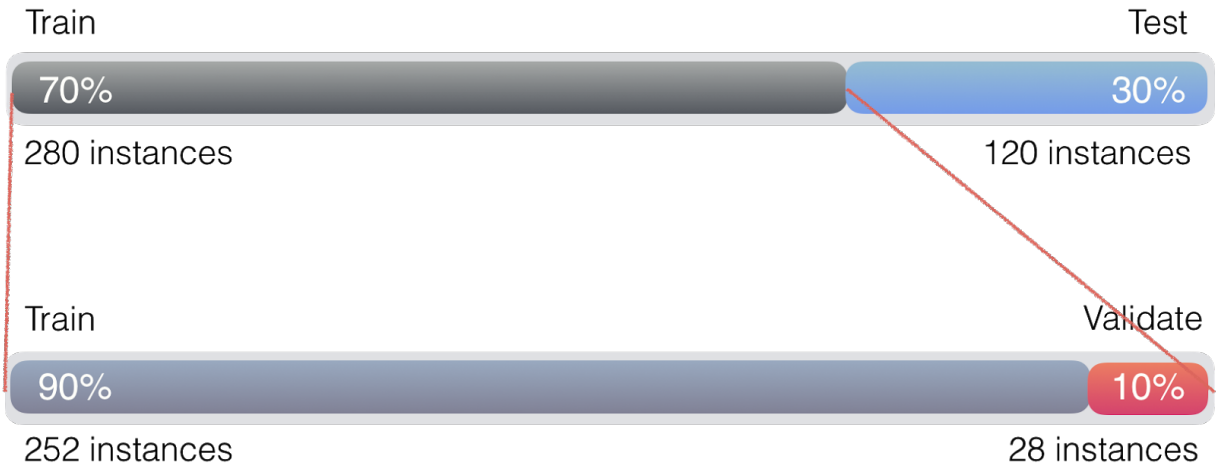
จำนวนข้อความ	จำนวนคำที่ unique ทั้งหมด	จำนวนข้อความแต่ละ intention
400	376	50



ภาพที่ 4 กราฟแสดงจำนวน information ทั้งหมดที่พบในชุดข้อมูล

Data preparation

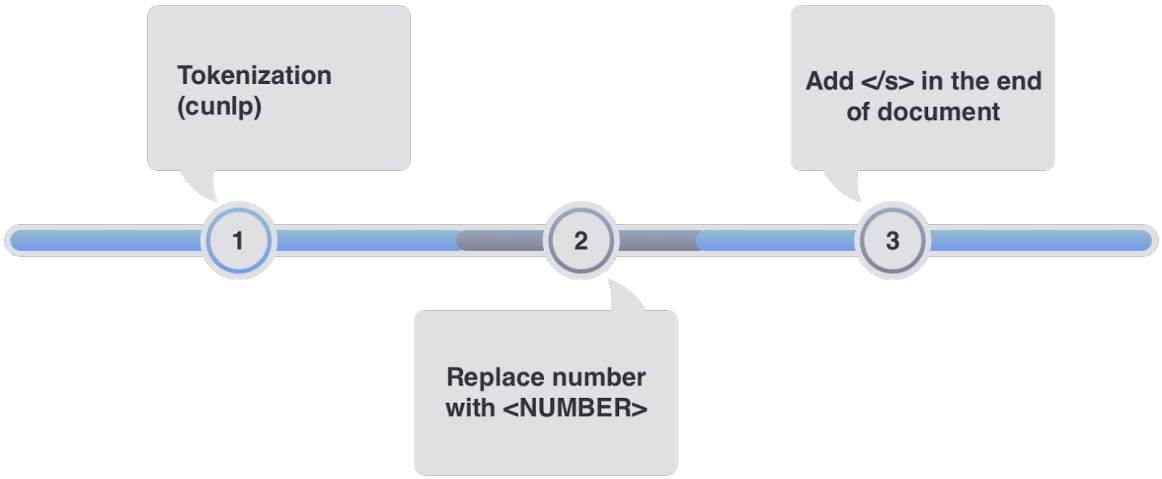
ในการแบ่งชุดข้อมูลสำหรับการเทรนโมเดล และทดสอบโมเดล ก่อนอื่นเราจะแบ่งข้อมูล Train Test ก่อนเป็นอัตราส่วน 70:30 และจากนั้น เราจะแบ่งข้อมูลสำหรับการทำ validate 10% จากข้อมูล train ทั้งหมด โดยจะมีจำนวนข้อมูลแต่ละส่วน ดังภาพด้านล่าง



ภาพที่ 5 แผนภาพอัตราส่วนของการแบ่งชุดข้อมูล

Data preprocessing

การเตรียมข้อมูลสำหรับเข้าโมเดลเพื่อทำการทำนาย intention จะแบ่งออกเป็น 3 ขั้นตอนดังนี้



ภาพที่ 6 ขั้นตอนในการประมวลผลข้อความนำเข้า

1. Tokenization (cunlp) - จะทำการตัดคำโดยใช้ library cunlp (Chulalongkorn University Natural Languages Processing Library (Beta))
2. Replace number with <NUMBER> - ขั้นตอนนี้จะทำเฉพาะส่วนของทำการ replace ตัวเลขทั้งหมดที่พบในข้อความ
3. Add </s> in the end of document - เราจะทำการต่อท้ายข้อความด้วย </s> เสมอ เพื่อช่วยในการบ่งบอกการจบของข้อความนั้น ๆ

ในส่วนของการสกัด information ที่จำเป็นต่อการตอบคำถามออกมานั้น เราจะผ่าน process เดียว คือการทำ Tokenization เท่านั้น

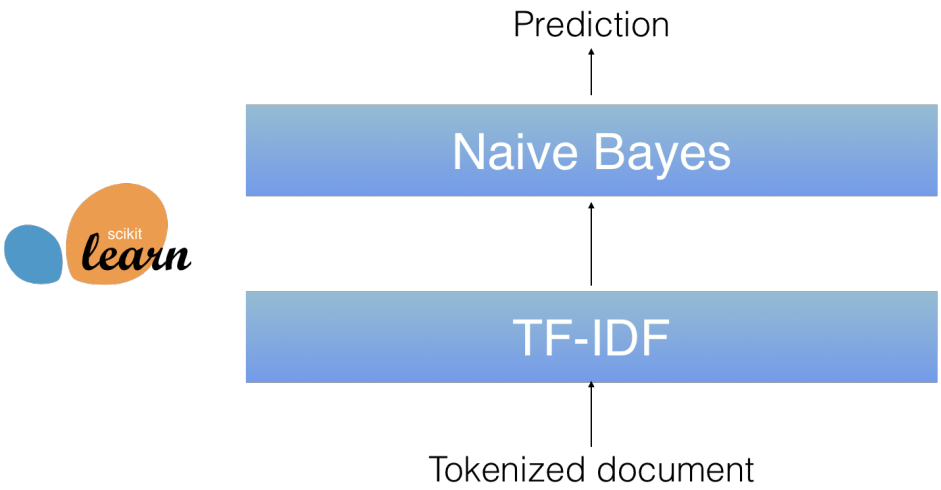
หลังจากที่ process ที่กล่าวมาข้างต้นจะได้ list ของคำที่ถูกตัดมาจากข้อความแต่ละข้อความ ซึ่งเราจะนำ list ของคำนี้ไป map หา index เพื่อเปลี่ยนจาก string เป็นตัวเลข โดยสำหรับคำที่ไม่เคยเจอหรือเป็น unknown word เราจะเปลี่ยนให้คำเหล่านี้เป็น index เดียวกัน

Model

Intention classification

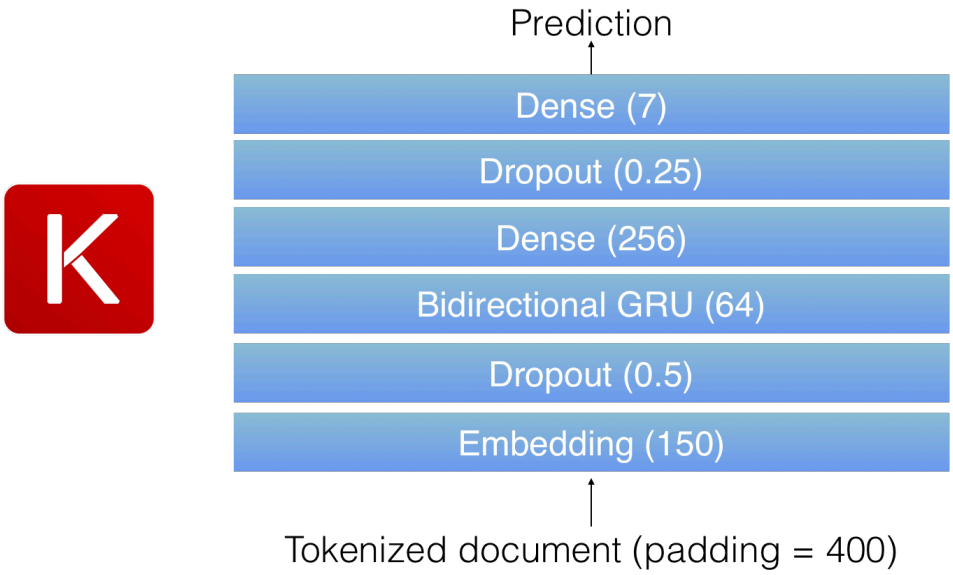
ในส่วนของการทำนาย intention เราจะมีการทดลอง 2 โมเดลคือ

1. Naive Bayes - เราจะใช้โมเดล Naive Bayes (Multinomial Naive Bayes) เป็นโมเดลหลักในการทำนาย โดยเราจะเปลี่ยนจาก list ของคำเป็น tf-idf เพื่อเป็น feature สำหรับโมเดล Naive Bayes



ภาพที่ 7 โครงสร้างของโมเดล intention classification แบบ Naïve Bayes

2. Bidirectional GRU - โมเดลนี้จะมีส่วนหลักคือการใช้ Bidirectional GRU เป็นหลัก โดยก่อนหน้าที่จะเข้า Bidirectional GRU เราจะนำ list ของคำที่เปลี่ยนเป็น index แล้ว เข้า Embedding layers โดย Embedding layers นี้ไม่ได้มีการใช้ pretrained weight ใดๆ

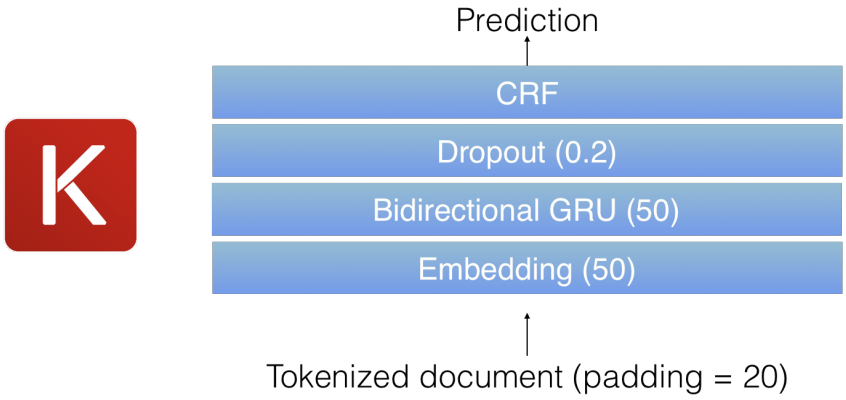


ภาพที่ 8 โครงสร้างของโมเดล intention classification แบบ Deep Neural Network

Information Extraction

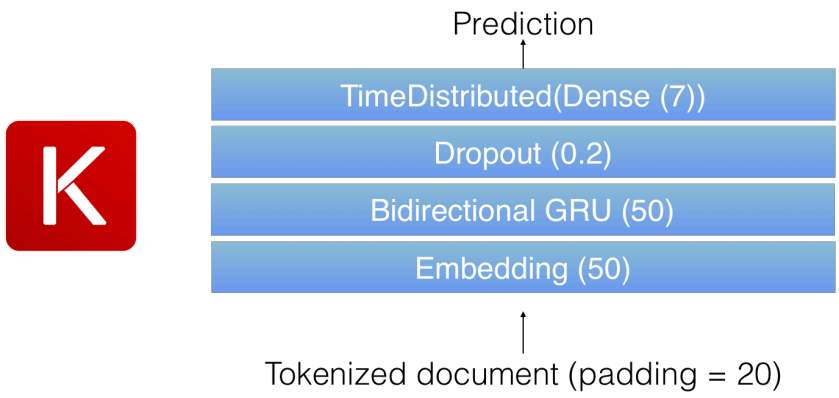
ในส่วนของการทำ information extraction เราจะมีการทดลอง 2 โมเดลคือ

1. CRF + Bidirectional GRU - โมเดลที่มีการใช้ Conditional random field (CRF) ผสมกับ Bidirectional GRU โดยเราจะมีการใช้ embedding เหมือนกับการทำ intention classification โดยเราจะไม่มีการใช้ pretrained weight



ภาพที่ 9 โครงสร้างของโมเดล information extraction แบบ Deep Neural Network + CRF

2. Bidirectional GRU - โมเดลนี้จะมีการใช้เฉพาะ Bidirectional GRU และมีการใช้ embedding ในลักษณะเดียวกัน



ภาพที่ 10 โครงสร้างของโมเดล information extraction แบบ Deep Neural Network

Evaluation

ในการวัดผล เราจะวัดผลโดยการแบ่งออกเป็น 2 ส่วนคือส่วนของโมเดล intention classification และ information extraction ซึ่งเราจะใช้ F1 macro และ F1 micro ในการวัดผลโมเดลแต่ละโมเดล โดยจะมีสมการดังตารางด้านล่าง

ตารางที่ 2 สมการของการวัดผลแบบ F1-macro และ F1-micro

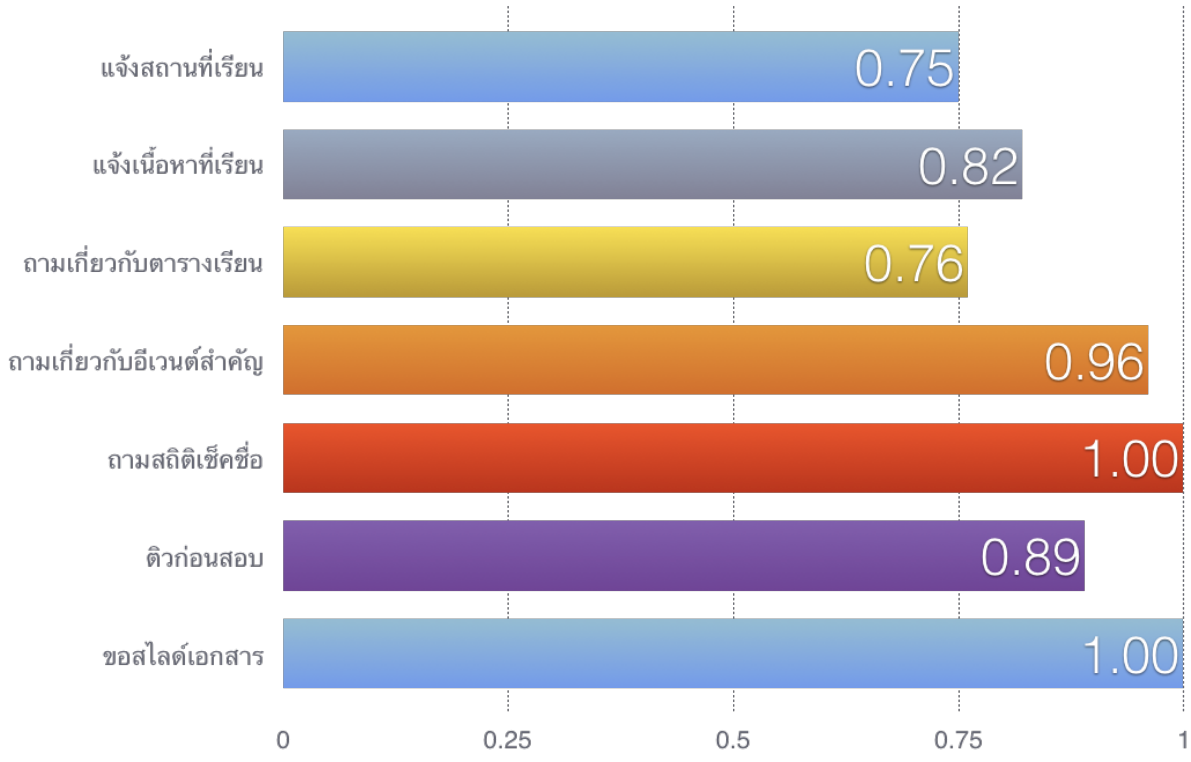
F1 macro	$\mathrm{MaF}_\beta = \frac{1}{\lvert C\rvert}\sum_{i=1}^{\lvert C\rvert} \mathrm{F}_{\beta,\,i}$
F1 micro	$\mathrm{MiF}_\beta = \frac{(\beta^2+1) \times \mathrm{MiPr} \times \mathrm{MiRe}}{\beta^2 + \mathrm{MiPr} + \mathrm{MiRe}}$

Intention classification

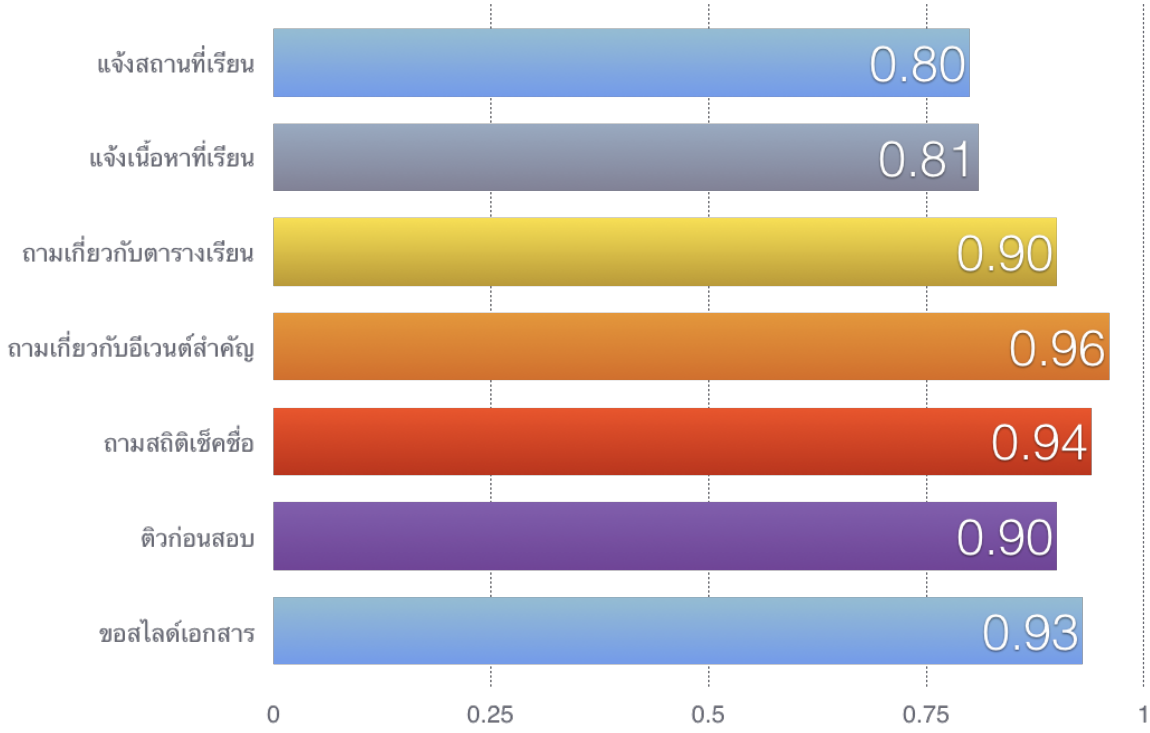
ส่วนของโมเดล intention classification นั้นจะทำการเปรียบเทียบ 2 โมเดลคือ Naive Bayes ที่ใช้ feature เป็น TFIDF และ Bidirectional GRU ซึ่งวัดผลออกมาได้ดังนี้

ตารางที่ 3 ค่า F1-macro และ F1-micro ของโมเดล intention classification ทั้ง 2 รูปแบบ

	F1 macro	F1 micro
Naive Bayes + TFIDF	**0.892**	**0.886**
Bidirectional GRU	**0.892**	**0.893**



ภาพที่ 11 กราฟของค่า F1 ของแต่ละคลาสที่ได้จากโมเดล Naive Bayes



ภาพที่ 12 กราฟของค่า F1 ของแต่ละคลาสที่ได้จากโมเดล Bidirectional GRU

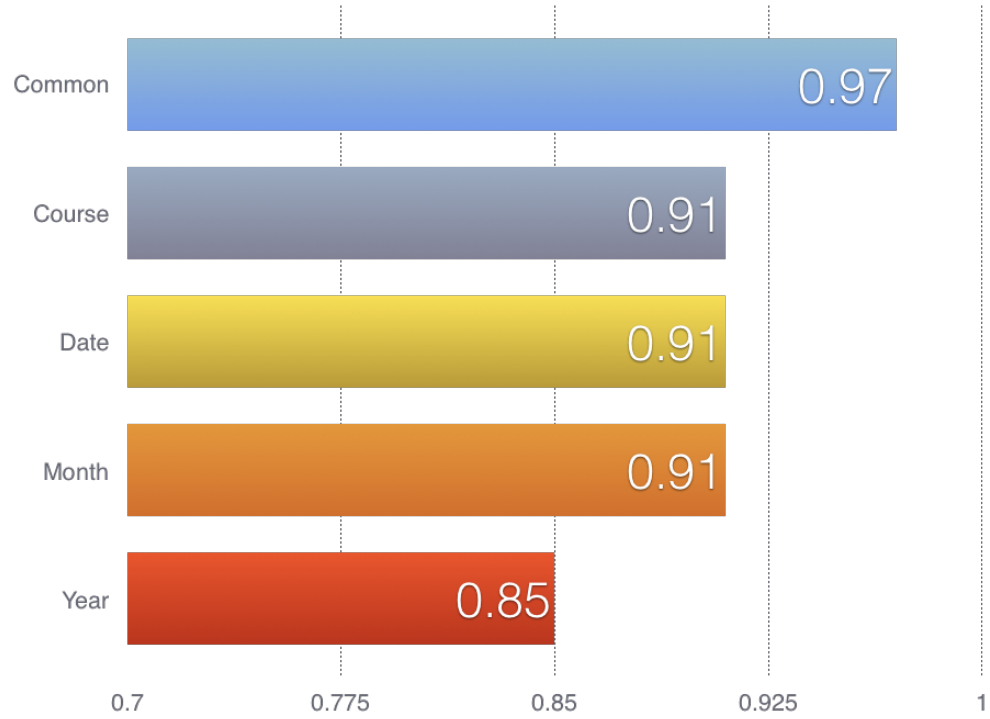
จากตารางและกราฟข้างต้นจะเห็นว่าในส่วนของ F1 micro จะพบว่า Bidirectional GRU จะได้ให้ผลที่ดีกว่า แต่ในส่วนของ F1 macro จะเห็นว่าออกมาได้ค่าเท่ากัน ซึ่งเมื่อสังเกตเพิ่มเติมจะเห็นว่า Bidirectional GRU นั้น จะให้ผลแต่ละคลาสใกล้เคียงกันมากกว่า Naive Bayes ที่มีบางคลาสที่ให้ผล F1 แค่ 0.75 แต่ในทางกลับกัน Bidirectional GRU จะผลของคลาสแย่ที่สุดเป็น 0.80

Information Extraction

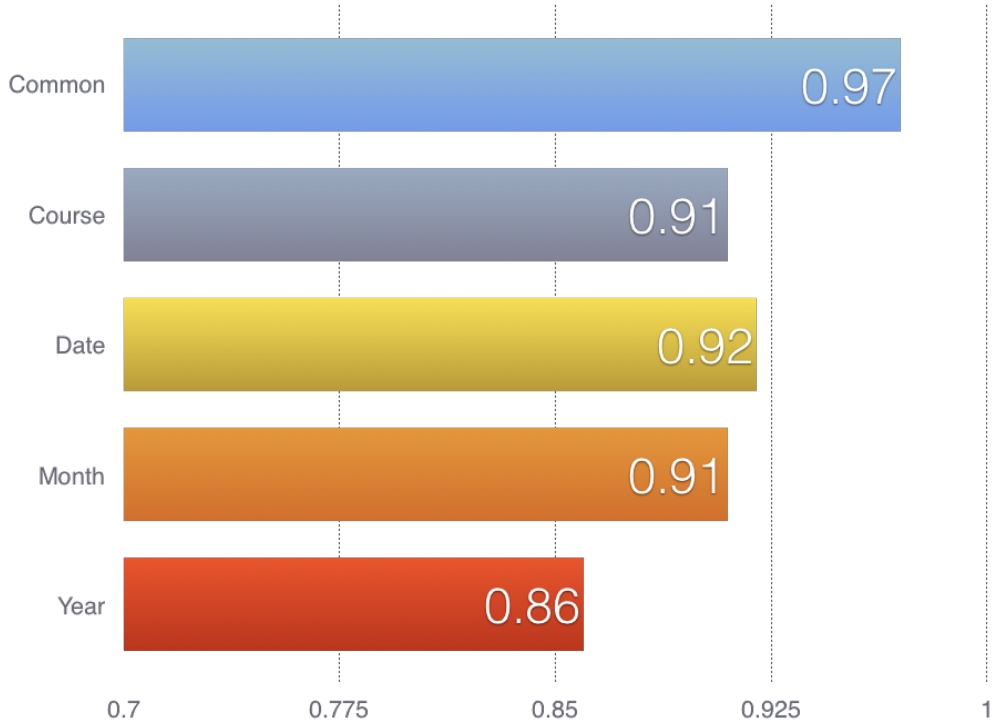
ส่วนของโมเดล information extraction นั้นจะทำการเปรียบเทียบ 2 โมเดลคือ Bidirectional GRU ที่ใช้ CRF และ ไม่ใช้ CRF ซึ่งวัดผลออกมาได้ดังนี้

ตารางที่ 4 ค่า F1-macro และ F1-micro ของโมเดล information extraction ทั้ง 2 รูปแบบ

	F1 macro	F1 micro
With CRF	**0.910**	**0.945**
Without CRF	**0.914**	**0.947**



ภาพที่ 13 กราฟของค่า F1 ของแต่ละคลาสที่ได้จากโมเดลที่มีการใช้ CRF



ภาพที่ 14 กราฟของค่า F1 ของแต่ละคลาสที่ได้จากโมเดลที่ไม่มีการใช้ CRF

จากตารางและกราฟข้างต้นจะเห็นว่า model ที่ไม่มีการใช้ CRF นั้นให้ประสิทธิภาพที่ดีกว่าในส่วนของ F1 macro และ F1 micro แต่เมื่อเปรียบเทียบผล F1 ของแต่ละคลาส จะเห็นว่าประสิทธิภาพนั้นดีกว่าแค่เพียงคลาส Date และ Year เท่านั้น

Issue

ในการทำงานของเรานั้น เราได้พบปัญหา และสิ่งที่ควรปรับปรุง ดังนี้

1. ตัวโมเดลนั้น overfitting กับชุดข้อมูลที่ใช้ในการเทรน เนื่องจากชุดข้อมูลนั้นมีขนาดเล็ก และมีความหลากหลายของภาษาที่ค่อนข้างต่ำ จึงทำให้ตัวโมเดลนั้นอาจจะไม่สามารถจัดการกับข้อความที่มีโครงสร้างนอกเหนือจากที่ข้อมูลที่ใช้เทรนได้ ดังนั้นเราจึงต้องการข้อมูลที่มีจำนวนมากขึ้น และหลากหลายมากขึ้นเพื่อจัดการปัญหานี้
2. ตัว Server ที่ใช้นั้น เป็น Server ที่เปิดให้ใช้ free ดังนั้นความสามารถในการประมวลผลโดยการใช้ model ต่าง ๆ อาจจะเกิดความล่าช้า และไม่สามารถรับการใช้งานของคนจำนวนมาก ๆ ได้ จึงอาจจะต้องมีการเปรียบ server ที่มีความสามารถในการประมวลผลมากกว่านี้
3. Keyword ของ information ที่ได้จากโมเดลมาจะต้องตรงกับ keyword ที่ต้องการใช้ในการ query database ต่าง ๆ อย่างเช่น ชื่อวิชา ที่จะไม่สามารถใช้ตัวย่อหรือชื่อย่อที่คนชอบเรียกกันได้ ซึ่งอาจจะแก้ไขโดยการเพิ่ม keyword ใน database ให้ search ได้มีประสิทธิภาพมากขึ้น
4. ขาดการวัดผลทั้ง flow ของ chatbot ซึ่งไม่สามารถทราบประสิทธิภาพที่แท้จริงของตัว chatbot ได้ ซึ่งเราอาจจะวัดผลโดยการให้ผู้ใช้เข้ามาทดลองใช้ chatbot และทำแบบประเมินว่า chatbot นั้นทำงานถูกต้อง ตรงตามที่ผู้ใช้ต้องการหรือไม่

Project Link

Project git repository: https://github.com/jrkns/nlp-bot-project

Project demo video: https://youtu.be/CZteSTKm6Wk