**สาระสำคัญ**

เนื่องจากในปัจจุบัน เอกสารความรู้ต่างๆ มากมาย ยังไม่ถูกนำมาจัดเก็บและจัดหมวดหมู่ให้เป็นระบบ ทำให้องค์ความรู้เหล่านั้นไม่สามารถนำมาใช้งานได้อย่างมีประสิทธิภาพ โดยในปัจจุบันนั้น การที่จะนำความรู้ที่มีอยู่เหล่านั้นมาจัดหมวดหมู่เพื่อใช้ในการสืบค้น จะต้องใช้มนุษย์เป็นผู้จัดการ ซึ่งจำเป็นต้องใช้แรงงานคนในการมาจำแนกหมวดหมู่เหล่านั้น โดยการพัฒนาการจัดการองค์ความรู้หรือ Knowledge management นี้ จะช่วยให้การนำความรู้ที่มีอยู่มาใช้ได้อย่างมีประสิทธิภาพมากขึ้น และช่วยส่งเสริมการพัฒนาประเทศชาติให้มุ่งไปสู่ความเป็นสังคมอุดมปัญญาได้ โดยที่ระบบ Knowledge management ที่มีอยู่ในปัจจุบันนั้น ยังไม่มีระบบที่สามารถนำเอาเอกสารไปทำการวิเคราะห์และ tag หมวดหมู่ของเนื้อหาได้โดยอัตโนมัติ โดยเฉพาะอย่างยิ่งในส่วนของภาษาไทย ซึ่งยังไม่มีนักพัฒนารายใดพัฒนาเทคโนโลยีในลักษณะนี้ออกมา ดังนั้น เราจึงคิดที่จะทำระบบที่สามารถนำเอาเอกสารหรือบทความต่างๆที่ถูกจัดเก็บไว้ในรูปของ PDF นำไปวิเคราะห์ข้อความและทำการสรุปว่า ข้อความนี้มีเนื้อหาที่เกี่ยวข้องกับเรื่องใดบ้าง และนำไปจัดเก็บลงไปยังฐานข้อมูล เพื่อให้สามารถทำการสืบค้นได้ โดยสิ่งที่ท้าทายสำหรับการทำโครงงานชิ้นนี้ก็คือ การที่ภาษาไทยไม่มีการแบ่งคำที่ที่ชัดเจนเหมือนภาษาอังกฤษที่มีการใช้ space คั่น ทำให้การวิเคราะห์รูปประโยคมีความยาก และการทำ machine learning ให้ได้ความแม่นยำในระดับที่สามารถนำไปใช้ได้จริงนั้น จะต้องใช้การเลือกใช้ algorithm และการปรับแต่งที่เหมาะสมกับข้อมูลที่นำมาใช้ จึงทำให้โครงการนี้มีความท้าทายในการดำเนินการ และเป้าหมายในการทำโครงการนี้ จะเป็นการพัฒนา model ที่สามารถนำข้อความจากเอกสารมา tag และทำการจัดหมวดหมู่ได้ และพัฒนา web application ที่สามารถสืบค้นข้อมูลที่ได้จาก model ข้างต้น เพื่อนำมาเป็น Proof of Concept ของ model ที่พัฒนาขึ้น

**คำสำคัญ**

**Machine Learning, Clustering: LDA(Latent Dirichlet allocation), Classification: One vs Rest, Big data, Spark, PDF to Text, Keyword extraction, Text Analysis**

**หลักการและเหตุผล**

ในยุคปัจจุบัน ที่มีการทำเอกสารในเรื่องต่างๆ ออกมาเป็นจำนวนมาก การทำ Knowledge management หรือการนำเอกสารข้อมูลเหล่านั้นมาจัดการให้เป็นระบบ นับเป็นเรื่องที่สำคัญมาก โดยเฉพาะในองค์กรหลายๆ แห่ง การมีระบบ knowledge management จะช่วยทำให้องค์กรนั้นๆ สามารถใช้งานองค์ความรู้ที่มีอยู่ได้อย่างมีประสิทธิภาพสูงสุด แต่ในปัจจุบัน เอกสารความรู้ต่างๆ ที่ถูกนำมาเผยแพร่อยู่นั้น มักจะอยู่ในรูปแบบของเอกสารในหน้ากระดาษ หรือเอกสารที่เป็นไฟล์ PDF ซึ่งยังไม่สามารถนำมาใช้งานได้อย่างมีประสิทธิภาพเท่าที่ควร เพราะว่า เอกสารเหล่านั้นมักจะมีข้อความอยู่มากมาย ที่เกี่ยวข้องกับเนื้อหาที่แตกต่างกัน แต่ว่าเมื่อผู้ที่ต้องการใช้งานความรู้เหล่านั้น ต้องการทำการหาเนื้อหาที่เฉพาะเจาะจงกับที่เขาสนใจในเอกสารนั้นๆ เขาก็ต้องทำการค้นหาด้วยตนเองโดยวิธีต่างๆ ไม่ว่าจะเป็นการไล่อ่านเนื้อหาทั้งหมดด้วยตนเอง ซึ่งใช้เวลามากในการอ่านและหาใจความสำคัญที่เขาต้องการ หรือใช้การค้นหา keyword ที่เขาต้องการด้วยวิธีต่างๆ เช่นการเปิดหาสารบัญ ซึ่งเอกสารบางฉบับก็ไม่มีสารบัญให้ หรือใช้การ search หา keyword ที่ต้องการ ซึ่งอาจจะเกิดการข้ามเนื้อหาในส่วนที่เกี่ยวข้องกับเรื่องที่ผู้ที่ค้นหาต้องการ แต่ไม่มี keyword ที่เขาใช้ค้นหาไปได้ ซึ่งสิ่งที่ได้กล่าวไปข้างต้นนั้น นับว่าเป็นปัญหาใหญ่ในการค้นคว้าหาข้อมูลเพื่อทำการศึกษาเป็นอย่างมาก เนื่องจากการที่ไม่มีระบบ knowledge management สำหรับเอกสารทั่วๆ ไปนั้น ทำให้แหล่งความรู้ที่สามารถนำมาสืบค้นได้นั้นลดลงเป็นอย่างมาก และทำให้ความรู้จำนวนมากถูกทิ้งร้างไว้ ไม่ได้ถูกนำมาใช้ให้เกิดประโยชน์ ดังนั้น ทางกลุ่มของเราจึงสนใจที่จะพัฒนา machine learning model ที่สามารถคัดแยกเนื้อหาในส่วนต่างๆ ในไฟล์เอกสาร และทำการ tag ข้อความเหล่านั้นได้โดยอัตโนมัติว่า เนื้อหาในส่วนนั้นๆ มีความเกี่ยวข้องกับเรื่องอะไรบ้าง และทำการจัดเก็บข้อมูลเหล่านั้นลงไปยังระบบฐานข้อมูล เพื่อให้สามารถทำการสืบค้นได้ง่ายและรวดเร็ว และทำให้การจัดการแหล่งความรู้ หรือ Knowledge management นั้น สามารถใช้งานกับเอกสารที่เป็นไฟล์ PDF ได้ ซึ่งส่งผลให้ความรู้ถูกนำไปใช้งานต่อ และเกิดการพัฒนาประเทศชาติในองค์รวมมากยิ่งขึ้น

**วัตถุประสงค์**

เพื่อสร้างเทคโนโลยีที่สามารถทำการวิเคราะห์ข้อความจากไฟล์ PDF และทำการจัดหมวดหมู่ของข้อความเหล่านั้นได้โดยอัตโนมัติ เพื่อให้เราสามารถนำเอกสารความรู้ต่างๆ ที่มีอยู่มากมายมาทำการจัดแบ่งกลุ่มและเก็บไว้ในฐานข้อมูลเพื่อใช้ในการสืบค้นได้ ซึ่งเทคโนโลยีนี้สามารถนำไปใช้งานในวงการการศึกษาได้เป็นอย่างดี โดยทำให้ผู้คนสามารถค้นหาความรู้จากแหล่งความรู้ที่กว้างขวางมากขึ้นโดยใช้เวลาที่น้อยลง และนำไปสู่การสร้างสังคมอุดมปัญญาต่อไปในอนาคต

**ปัญหาหรือประโยชน์ที่เป็นเหตุผลให้ควรพัฒนาโปรแกรม**

ในปัจจุบันนั้น มีโปรแกรมที่ถูกพัฒนาโดยมหาวิทยาลัยสแตนฟอร์ด ที่ชื่อว่า Stanford Deepdive ที่ทำการวิเคราะห์ข้อความและ tag เนื้อหาเหล่านั้นได้โดยการใช้ Machine Learning แต่ว่าโปรแกรม Stanford Deepdive ที่กล่าวมานั้น ถูกพัฒนาขึ้นสำหรับการวิเคราะห์เอกสารในภาษาอังกฤษเป็นหลัก ซึ่งภาษาไทยที่มีรูปแบบของประโยค การจัดเรียงคำ การวางตำแหน่งคำ,ตัวอักษร และอื่นๆ ที่แตกต่างจากภาษาอังกฤษเป็นอย่างมาก ทำให้การทำระบบ tag เอกสารอัตโนมัติสำหรับภาษาไทยนั้น ไม่สามารถใช้ Stanford Deepdive ได้ และการ tag หมวดหมู่ให้กับข้อความจำนวนมากโดยใช้มนุษย์ในการจัดการนั้น จะเป็นการเสียเวลาไปเป็นจำนวนมาก ทำให้ทางกลุ่มของเราสนใจที่จะพัฒนาโปรแกรมในลักษณะคล้ายกันกับ Stanford Deepdive ที่สามารถนำมาใช้กับภาษาไทยได้ เพื่อให้เอกสารต่างๆ ที่เป็นภาษาไทยนั้น ถูกนำมาใช้ประโยชน์ และนำมาศึกษาต่อได้อย่างมีประสิทธิภาพ

**เป้าหมายและขอบเขตของโครงการ**

ทำการพัฒนา machine learning model ที่สามารถรับข้อมูลจากไฟล์ PDF ที่เป็นภาษาไทย และทำการ tag หมวดหมู่ให้กับแต่ละข้อความเหล่านั้น และทำการพัฒนา web application เพื่อที่จะใช้ในการสืบค้นข้อมูลที่ทำการ tag มาแล้วได้ โดยมีขอบเขตในการทำคือ เนื้อหาที่สามารถนำมาให้ตัว model ทำการจัดหมวดหมู่ได้นั้น ต้องเป็นเนื้อหาที่เกี่ยวข้องกับเรื่องที่ตัว model เคยเรียนรู้มาแล้ว

**รายละเอียดของการพัฒนา**

**7.1 Story board**

เมื่อต้องการที่จะค้นหาเอกสารที่เกี่ยวข้องกับเรื่องใดเรื่องหนึ่งขึ้นมาใช้งาน ผู้ใช้จะสามารถค้นหาข้อมูลได้อย่างรวดเร็วด้วย platform ที่ทางกลุ่มพัฒนาขึ้น โดย platform ที่พัฒนาขึ้นจะแบ่งออกได้เป็น 2 ส่วนหลักๆ

* ส่วนของผู้พัฒนาและผู้ดูแลระบบ โดยผู้พัฒนาจะทำการเตรียมเอกสารที่เป็นไฟล์ PDF เพื่อทำการเตรียม machine learning model ที่สามารถทำการ tag เอกสารได้โดยอัตโนมัติ โดยมีการผู้เชี่ยวชาญเฉพาะมาช่วยในการระบุคำสำคัญของเนื้อหาในเรื่องต่างๆ ที่จะนำมาใช้ในการเตรียม model เช่น ถ้าต้องการให้ตัว model สามารถทำการจำแนกเนื้อหาที่เกี่ยวข้องกับเรื่อง “การดึงความสนใจนักเรียน” ผู้พัฒนา/ผู้ดูแลจะต้องเตรียมเอกสารที่มีเนื้อหาที่เกี่ยวข้องกับเรื่องการดึงความสนใจของนักเรียนไว้ และให้ผู้เชี่ยวชาญช่วยระบุว่า มีคำใดบ้างที่สามารถระบุได้ว่า ข้อความนี้มีความเกี่ยวข้องกับเรื่องการดึงความสนใจของนักเรียน และนำไปทำการเตรียม model
* ส่วนของผู้ใช้งาน ผู้ใช้สามารถเข้ามาใช้งานผ่าน web application ที่ทางกลุ่มพัฒนาขึ้นมา แล้วทำการค้นหาเนื้อหาที่เกี่ยวข้องกับสิ่งที่ผู้ใช้ต้องการ แล้วเนื้อหาส่วนนั้นก็จะปรากฏขึ้นมา และมีไฟล์เอกสารนั้นให้ผู้ใช้สามารถ download ไปอ่านได้ ยกตัวอย่างเช่น ครูสมศรีต้องการที่จะหาข้อมูลเรื่อง “การดึงความสนใจนักเรียน” เพื่อนำไปเตรียมการเรียนการสอนสำหรับชั้นเรียน สิ่งที่คุณครูต้องทำก็คือ ค้นหาด้วยคำว่า “ดึงความสนใจนักเรียน” ในหน้าเว็บ แล้วเว็บก็จะทำการแสดงผลย่อหน้าที่เกี่ยวข้องกับการดึงความสนใจนักเรียน และ tag ที่เกี่ยวข้องกับย่อหน้านั้นๆ โดยแต่ละย่อหน้าก็จะมี tag ที่เกี่ยวข้องเป็นของตัวเอง และมีลิงค์สำหรับดาวน์โหลดเอกสารที่มีข้อความนั้นอยู่ให้คลิกเพื่อดาวน์โหลดได้

โดย platform นี้จะมีจุดเด่นที่เราสามารถนำ model นี้ ไปประยุกต์ใช้กับหัวข้ออี่นๆได้ โดยไม่จำเป็นต้องออกแบบโปรแกรมใหม่ทั้งหมด เพียงแค่เตรียมเอกสารที่เกี่ยวข้องและคำสำคัญของหัวข้อนั้นๆ และให้ระบบทำการเรียนรู้ด้วยตัวเอง



**7.2 เทคนิคหรือเทคโนโลยีที่ใช้**

* Natural Language Processing เป็นอัลกอริทึมที่ใช้เพื่อวิเคราะห์ภาษาธรรมชาติของมนุษย์ ด้วยการรับค่าที่เป็นประโยค จากนั้นทำการวิเคราะห์ประมวลผลภาษาธรรมชาติเพื่อให้ได้ผลลัพธ์ในการใช้งานต่อ เช่นการทำ Word Segmentation จะเป็นการแบ่งคำออกจากกันให้เป็นคำที่มีความหมายเพื่อนำไปใช้งานต่อโดยวิธีการเหล่านี้จำเป็นต้องทำงานร่วมกับฐานความรู้อื่นด้วย ดังตัวอย่างการทำ Word Segmentation จำเป็นต้อง Dictionary Database เพื่อช่วยในการแบ่งคำ และการวิเคราห์ภาษาธรรมชาติยังรวมถึงการทำการประมวลผลความหมายของคำต่าง เช่นการตรวจสอบ Sentiment ของคำต่างๆ อีกด้วย
* Term frequency – Inverse document frequency (TF-IDF) เป็นวิธีทางสถิติที่จะทำการตรวจสอบคำต่างๆในบทความเพื่อนำไปเปรียบเทียบกับบทความทั้งหมด เพื่อหาอัตราส่วนว่าคำๆนี้มีความสำคัญต่อบทความโดยรวมแค่ไหน โดย TF-IDF จะแบ่งขั้นตอนเป็น 2 ส่วนคือ Term frequency โดยในขั้นตอนนี้นั้นจะทำการนับจำนวนครั้งที่คำต่างๆปรากฎในบทความหนึ่งๆ และการทำ Inverse document frequency โดยในขั้นตอนนี้จะเป็นนำคำต่างๆในบทความมาเปรียบเทียบกับบทความทั้งหมดและคำนวณหาค่าน้ำหนักความสำคัญนั้นๆ จากบทความทั้งหมด โดยการทำ TF-IDF นั้น สามารถใช้ประโยชน์ในการหาคำสำคัญในบทความต่างๆซึ่งสามารถนำไปประยุคใช้ได้อย่างหลากหลายเช่นการทำ Search engine หรือการทำ Text Summarization
* Machine Learning: Classification เป็นวิธีในการจำแนกผลลัพธ์จากข้อมูล input โดยจะแบ่งขั้นตอนการใช้งานเป็น 2 ขั้นตอนคือการ Training Model และ Testing Model โดย การ Training Model นั้น จะทำการรับ Input จำนวนมากเพื่อสร้าง Model จาก Input เหล่านั้น และการทำ Testing Model จะเป็นการนำ Input ที่เราต้องการใช้ มาผ่าน Training Model ในขั้นตอนข้างต้น เพื่อให้ระบบประมวลผลและจำแนกแผลลัพธ์ออกมา โดยการทำ Classification มีวิธีการที่หลากหลาย ซึ่งในที่นี้ เราจะเลือกใช้ Multiclass Classification เช่น One-vs-Rest

**7.3 เครื่องมือที่ใช้ในการพัฒนา**

* Hadoop Distributed File System (HDFS) เป็นระบบการจัดเก็บข้อมูลที่ออกแบบมาสำหรับการจัดการข้อมูลขนาดใหญ่ (Big data) โดย HDFS ถูกออกแบบมาสำหรับระบบที่มีคอมพิวเตอร์หลายๆ ตัวช่วยกันประมวลผล และ HDFS จะเหมาะกับการทำงานในลักษณะ “Write once, Read many” หรือข้อมูลที่เน้นการอ่านข้อมูลมากกว่าการเขียน,แก้ไข โดยการทำงานของ HDFS จะแบ่งคอมพิวเตอร์ที่อยู่ในระบบเป็น 2 ส่วนด้วยกัน ได้แก่ Name node และ Data node โดยเมื่อต้องการที่จะจัดเก็บไฟล์ลงในระบบ HDFS ไฟล์ที่เข้ามาจะถูกแบ่งออกเป็นหลายๆ ส่วนและแบ่งไปเก็บไว้ยัง data node แต่ละเครื่อง ส่วน name node จะมีหน้าที่เก็บข้อมูลว่า ไฟล์นั้นๆ ถูกแบ่งออกเป็นกี่ส่วน แต่ละส่วนถูกเก็บไว้ที่คอมพิวเตอร์เครื่องใดบ้าง และเมื่อต้องการนำไฟล์ออกมาใช้งาน name node จะทำการสั่งงานให้ data node ทุกเครื่องที่มีส่วนของไฟล์นั้นๆ ทำการอ่านส่วนของไฟล์ที่เก็บไว้ส่งไปยัง name node และตัว name node จะทำการรวมส่วนของไฟล์ทั้งหมดที่ได้รับมาให้เป็นไฟล์เดียวกัน เพื่อให้สามารถใช้งานไฟล์นั้นได้ต่อไป
* โปรแกรมสำหรับการแปลงไฟล์ในรูปแบบ PDF ให้เป็น text file โดยทางกลุ่มเลือกใช้ PDFMiner ซึ่ง PDFMiner เป็น Python API ที่ใช้สำหรับการดึงข้อมูลต่างๆออกมาจาก PDF Document เช่นตัวอักษรในภาษาต่างๆ เช่น ไทย อังกฤษ จีน และอื่นๆ หรือสามารถดึงภาพออกจาก PDF ได้ โดยสำหรับโปรเจคนี้จะเน้นที่การดึงข้อความออกจาก PDF Document เพื่อสำหรับนำไป preprocess ต่อ ซึ่งฟังก์ชั่นที่ใช้ในการดึงข้อความออกมานั้นคือ PDF2TXT โดยคำสั่งต่างๆของฟังชั่นนี้ สามารถเลือก page number, ชนิดของ output (text,tag,xml), ขนาดของ box ของคำใน pdf เป็นต้น
* โปรแกรม Spark ML เป็น library ที่มีอยู่ในโปรแกรม Apache Spark ซึ่ง Spark ML เป็น libraryที่ใช้ทำ Machine Learning โดยที่สามารถทำงานแบบขนาน (Parallel programming) ได้ ซึ่ง Apache Spark เป็น engine สำหรับการทำการประมวลผลข้อมูลขนาดใหญ่ (Big data processing) ที่สามารถทำงานได้อย่างรวดเร็ว เนื่องจากใช้การประมวลผลในหน่วยความจำหลัก (In-memory processing) ทำให้การเข้าถึงข้อมูลทำให้รวดเร็วมากขึ้น
* โปรแกรม LexTo เป็นโปรแกรมที่ถูกพัฒนาด้วยภาษา Java โดยโปรแกรมนี้สามารถใช้ในการแบ่งคำต่างๆในภาษาไทยจากโปรโยคให้กลายเป็นคำซึ่งแบ่งด้วย delimiter ซึ่งคำต่างๆที่ใช้ในการแบ่งนั้น จะมี Dictionary ที่จะทำการเก็บคำทั้งหมดเอาไว้ แล้วโปรแกรมจะนำมาเปรียบเทียบเพื่อแบ่งคำตามที่ Dictinary ได้กำหนดไว้
* โปรแกรม Apache Impala เป็นโปรแกรมจัดการฐานข้อมูลแบบ Open source ที่ถูกออกแบบมาให้ใช้งานร่วมกับ Hadoop ecosystem โดย Impala จะเหมาะกับการเก็บข้อมูลที่ต้องการนำมาวิเคราะห์แบบรวดเร็ว เนื่องจากตัวโปรแกรมมี latency ต่ำและมี throughput ที่สูง และยังมีความสามารถในการเพิ่มประสิทธิภาพของระบบได้ง่าย (Scalable)

**7.4 รายละเอียดโปรแกรมที่จะพัฒนา**

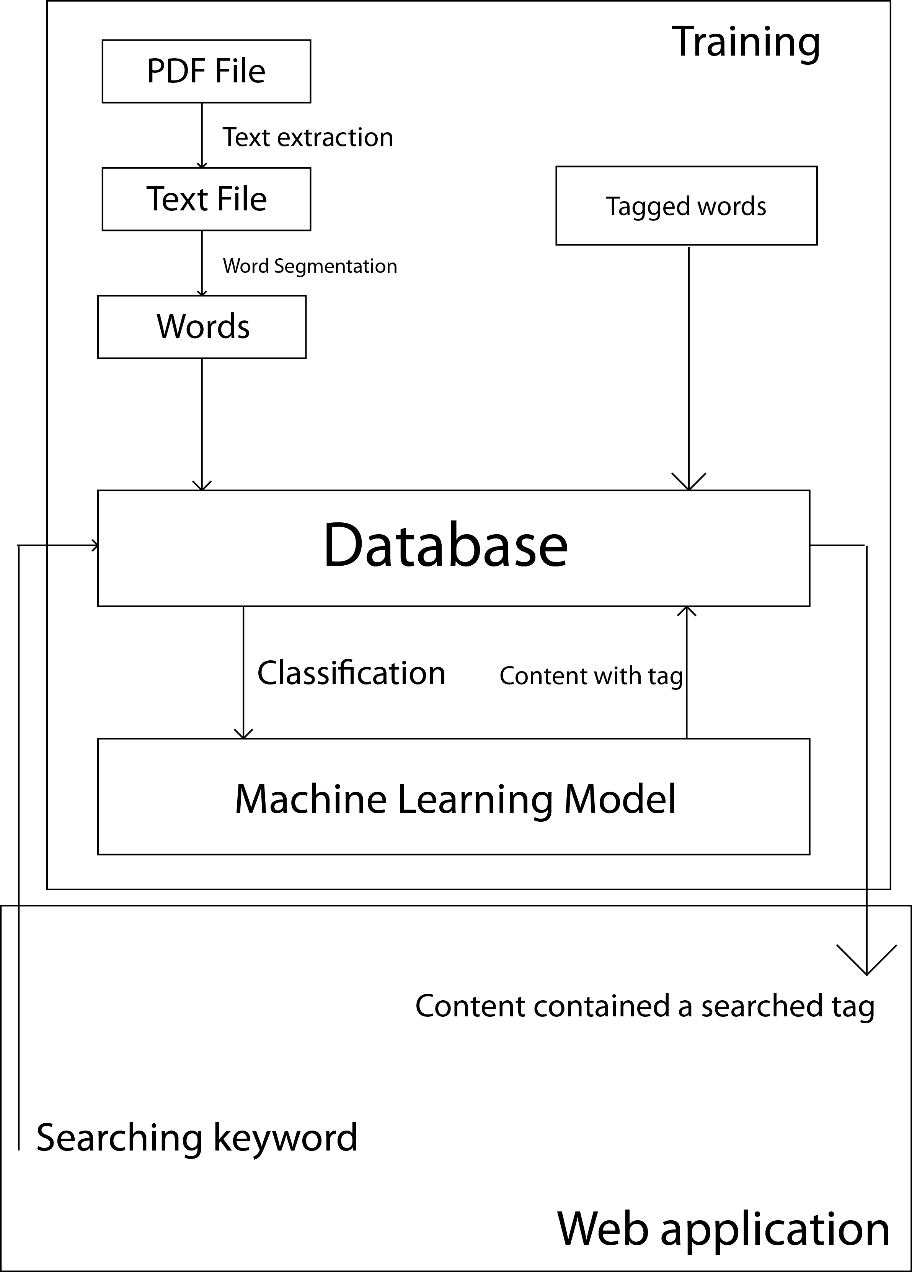
**Input/Output Specification**

ส่วนของผู้ดูแลระบบที่นำข้อมูลเข้าไปให้ระบบเรียนรู้

* Input เป็นเอกสารไฟล์ PDF ภาษาไทย ที่มีเนื้อหาเกี่ยวข้องกับสิ่งที่ machine learning model ได้เรียนรู้ไว้แล้ว และคำที่ถูก label tag ไว้โดยผู้เชี่ยวชาญแล้ว
* Output เป็น model ที่ได้จากการสร้างโดย input ข้างต้น และเก็บข้อมูลเนื้อหาและ tag ลงไปยัง database

ส่วนของผู้ใช้งาน

* Input เป็น tag ที่ต้องการค้นหา
* Output เป็นย่อหน้าสำคัญที่เกี่ยวข้องกับ tag นั้นๆ และ tag ทั้งหมดของย่อหน้าที่แสดง และลิงค์ให้ download PDF file ที่มีย่อหน้านั้นอยู่



**Functional Specification**

**Text Processing**

* ทำการแปลง PDF ให้เป็น text fileโดยการใช้ PDFMiner
* ทำการ preprocess text file ที่ได้ โดยการตัด tagged PDF ที่ติดมาหลังจากการทำการแปลง และทำการแก้ภาษาไทยที่ผิดพลาด เช่น สระ อำ (อ า เป็น อำ)
* นำเอา text file ที่ได้มาทำการตัดคำ (word segmentation) โดยใช้ LexTo
* ทำการนำ stop word ออก เพื่อลดคำที่ไม่จำเป็นออก
* ทำการ word count และทำ word to vector ของแต่ละ document

**Classification: Training model**

* นำ keyword ที่ได้จากการทำ Clustering มา label
* นำ keyword และ label ที่ได้มา training classifier model
* ใช้ Multiclass and multilabel algorithms (One-Vs-The-Rest)

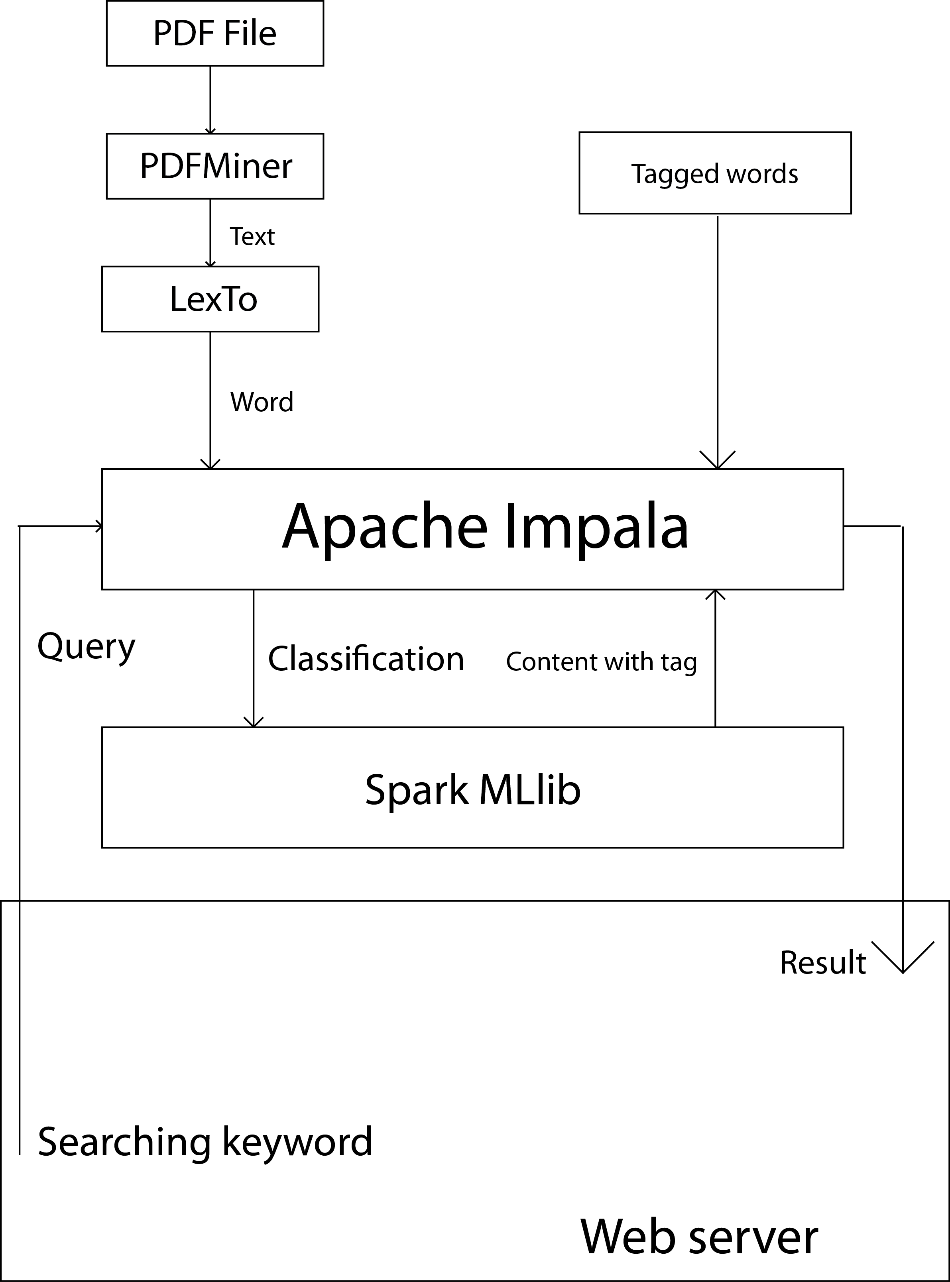
**Classification: Test model**

* นำ keyword ที่ได้จากการ Clustering ของ test document มาทำการ Classification โดยใช้ training model ที่ได้จากขั้นตอนการทำ Training model เพื่อบอก tag ของ output ออกมา
* ใช้ Multiclass and multilabel algorithms (One-Vs-The-Rest)

**Database** แบ่งเป็น 3 ส่วนหลักๆ คือ

* Keyword database เก็บข้อมูลคำที่เป็น keyword ต่างๆ โดยเก็บคำและความถี่ที่ปรากฏของคำนั้นๆ
* Document database เก็บข้อมูลเอกสาร โดยเก็บชื่อเอกสาร เนื้อหา และ tag
* Label database เก็บข้อมูลว่าแต่ละ label มีคำอะไรบ้าง

**Architecture**

****

**7.5 ขอบเขตและข้อจำกัดของโปรแกรมที่พัฒนา**

- เนื่องจากภาษาไทยเป็นภาษาที่มีความซับซ้อนสูง ทั้งทางด้านตัวอักษร ที่มีสระบน-ล่าง และทางด้านรูปประโยคที่ไท่ทีความแน่นอน ทำให้การเขียนโปรแกรมที่สามารถประมวลผลภาษาไทยได้อย่างสมบูรณ์แบบจึงเป็นเรื่องยาก ทำให้ความแม่นยำในการ tag และเลือกย่อหน้าที่มีความสำคัญกับเรื่องที่เลือก อาจจะต่ำกว่าการใช้งานกับภาษาอังกฤษ ที่มีรูปประโยคที่แน่นอนกว่า ทำให้สามารถใช้การดูรูปประโยคเข้ามาช่วยเสริมความหมายของคำได้ ซึ่งเป็นสิ่งที่ทำได้ยากมากในภาษาไทย

- ข้อมูลที่จะนำไปเข้าระบบ machine learning เพื่อให้ระบบทำการเรียนรู้ด้วยตนเองนั้น จะต้องใช้มนุษย์เป็นตัวช่วยในการกำหนดข้อมูลก่อนในเบื้องต้น เพราะฉะนั้น ถ้าเราต้องการให้ระบบเรียนรู้เนื้อหาเรื่องใหม่ๆ จะต้องมีการใช้ผู้เชี่ยวชาญที่เกี่ยวข้องกับเรื่องที่จะให้ระบบเรียนรู้มาช่วยทำการ label คำสำคัญก่อนที่จะนำข้อมูลเข้าไปในระบบ ดังนั้น ถ้าเกิดเราไม่สามารถหาผู้ที่จะมาระบุคำสำคัญให้ได้ เราก็จะไม่สามารถทำให้ระบบเรียนรู้หัวข้อใหม่ๆ ได้

**บรรณานุกรม**

* มาเริ่มเรียนรู้ Hadoop กันหน่อย, <http://www.somkiat.cc/start-with-hadoop/> (Accessed 2016-9-23)
* Apache Spark, <http://spark.apache.org> (Accessed 2016-9-23)
* Pdfminer, <http://euske.github.io/pdfminer/index.html>
* PDFMiner, http://www.unixuser.org/~euske/python/pdfminer/
* One-vs-Rest classifier, <https://spark.apache.org/docs/latest/ml-classification-regression.html#one-vs-rest-classifier-aka-one-vs-all>
* Lexto , http://www.sansarn.com/lexto/
* Latent Dirichlet allocation(LDA), <https://spark.apache.org/docs/1.6.0/ml-clustering.html#latent-dirichlet-allocation-lda>
* Impala, http://impala.apache.org/