การพัฒนาไปป์ไลน์วิเคราะห์สำหรับ Metagenome

จัดทำเพื่อ
ศูนย์พันธุวิศวกรรมและเทคโนโลยีชีวภาพแห่งชาติ
สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ

จัดทำโดย
นายวัชระ ศิริเนาวกุล

๒๗ ธันวาคม ๒๕๖๑

เอกสารส่งมอบรายงานจ้างพัฒนา

1. สคริปสำหรับวิเคราะห์ข้อมูล Microbial Metagenome
2. ฐานข้อมูล Kegg Prokaryotic Gene
3. การวิเคราะห์ Microbiome ด้วย Network Visualization

เรื่องที่ส่งมอบ

1) สคริปสำหรับวิเคราะห์ข้อมูล Microbial Metagenome
2) ฐานข้อมูล Kegg Prokaryotic Gene
3) การวิเคราะห์ Microbiome ด้วย Network Visualization

สื่อประกอบการส่งมอบ

1. CD ที่รวบรวมโปรแกรมที่ถูกพัฒนาในโปรเจคนี้ ซึ่งประกอบด้วย 3 โฟล์เดอร์ย่อยดังนี้
    1.1. metagenomic_pipeline/ สำหรับ ระบบไปป์ไลน์สำหรับวิเคราะห์ข้อมูล Metagenome ของ microbiome
    1.2. keggdb/ สำหรับ ฐานข้อมูล Kegg Prokaryotic Gene
    1.3. docker_species/ สำหรับ การวิเคราะห์ Microbiome ด้วย Network Visualization
2. ข้อมูลทั้งหมดที่อยู่ใน CD จะถูกเก็บไว้ใน HPC ของ BIOTEC ด้วยบน PATH sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy project/

* เนื่องด้วยฐานข้อมูล Kegg Prokaryotic Gene มีขนาด 24.2 GB ซึ่งมีขนาดใหญ่เกินขนาดของ CD ผู้เขียนจึงขอเก็บฐานข้อมูล Kegg Prokaryotic Gene นี้ไว้บน HPC ของ BIOTEC ที่เดียวที่ PATH sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy project/keggdb/ โดยเมื่อเข้าไปใน PATH ที่ระบุ จะพบ 2 โฟลเดอร์คือ

- download/ (ขนาด 8.2 GB) คือข้อมูลที่ดาวโหลดมาจากเซิฟเวอร์ของ Kegg
- keggdb/ (ขนาด 15.9 GB) คือข้อมูลจาก download/ ที่ถูกโปรเซสแล้วไว้ใช้สำหรับ Blast

สารบัญ

บทนำ

คำศัพท์สำคัญ

Genome คือ ข้อมูลทางพันธุกรรมทั้งหมดที่จำเป็นต่อการสร้างและดำรงชีพของสิ่งมีชีวิต

Metagenome การศึกษาสารพันธุกรรมทั้งหมดของสังคมจุลินทรีย์ในสิ่งแวดล้อมหนึ่งๆ เพื่อหาความสัมพันธ์ของกลุ่มประชากร และวิเคราะห์ถึงองค์ประกอบในความหลากหลายของระบบนิเวศ สามารถแบ่งได้เป็น Function-based metagenomics และ Sequence-based metagenomics
ที่มา http://www.smj.ejnal.com/e-journal/showdetail/?show_preview=T&art_id=1605

Microbiome คือ กลุ่มจุลินทรีย์ที่อยู่ ณ บริเวณใดบริเวณหนึ่ง

Microbial culture หรือ การเพาะเชื้อ คือ การนำเชื้อมาเพาะเลี้ยงที่สภาพแวดล้อมที่เหมาะสมต่อการเจริญเติบโต Culture ยังมีอีกความหมายว่า เชื้อที่ถูกเพาะเลี้ยงอีกด้วย
ที่มา http://haamor.com/th/การตรวจเพาะเชื้อ

Genome sequencing คือ กระบวนการในการหาลำดับของ Nucleotide (A/T/C/G) เพื่อให้ได้ข้อมูล Genome
ที่มา http://www.genomenewsnetwork.org/resources/whats_a_genome/Chp2_1.shtml

Blast ย่อมาจาก Basic Local Alignment Search Tool คือเครื่องมือที่ใช้ค้นหา Sequence เช่น DNA, Protein จากฐานข้อมูล ที่มีความคล้ายคลึงกับ สาย Sequence ค้นหาที่สุด

คำศัพท์จาก NCBI

เพื่อให้ง่ายในการสื่อสารระหว่างนักวิจัย NCBI ได้ให้คำจำกัดความของศัพท์ต่างๆ และคำศัพท์เหล่านี้ได้ถูกใช้ในฐานข้อมูลของ NCBI ด้วย คำศัพท์ที่จำเป็นในรายงานนี้ ได้แก่

BIoproject คือ กลุ่มข้อมูลที่ถูกสร้างโดยมีเป้าหมายเดียว และจากองค์กรเดียว โดยมีเลข Bioproject accession number (PRJNAxxxxxx) เป็นเลข ID
ที่มา https://www.ncbi.nlm.nih.gov/bioproject/docs/faq/

Biosample โดยมากถูกอ้างอิงถึง วัตถุชิ้นหนึ่งๆ ซื่อประกอบด้วย วัสดุทางชีวภาพ และมีข้อมูลที่เรียกว่า assay ประกอบด้วย เช่น blood samples, cell cultures โดย Biosample จะเป็นส่วนประกอบของ Bioproject
ที่มา https://www.ebi.ac.uk/training/online/course/biosamples-quick-tour/what-biosamples

Sequence Read Archive (SRA) คือ ชุดข้อมูลพันธุกรรมที่เรียกว่า short reads ที่ได้จาก High throughput sequencing ซึ่งส่วนใหญ่มีขนาดเล็กกว่า 1,000 base pairs โดยข้อมูล SRA ชุดหนึ่งสามารถเข้าเรื่อง High throughput sequencing ได้หลายรอบ ซึ่งแต่ละรอบจะถูกเรียกว่า Sequence Read Run (SRR) และ ข้อมูล SRA ใดๆ สามารถประกอบด้วย Biosample หลายๆชนิดเข้าด้วยกัน
ที่มา https://en.wikipedia.org/wiki/Sequence_Read_Archive

การวิเคราะห์ Metagenome โดยใช้เครื่องมือของ NCBI

ในอดีต การศึกษาลำดับรหัสพันธุกรรมของสิ่งมีชีวิต นักวิทยาศาสตร์จะทำการเพาะเลี้ยงสิ่งมีชีวิตแยกทีละชนิด และนำไปหาลำดับรหัสพันธุกรรม เพื่อทำการทดลอง แต่วิธีการนี้มีปัญหาคือ ไม่สามารถแสดงผลลัพธ์ตามสภาพจริงที่มันอาศัยอยู่ เมื่อเทคโนโลยีก้าวหน้าขึ้น โดยเฉพาะเทคโนโลยี Genome sequencing มีความเร็วสูงขึ้นและราคาถูกลง นักวิทยาศาตร์จึงมีการปรับปรุงวิธีการทดลองมาเป็น Metagenome

เพื่อให้เข้าใจภาพรวมของโปรเจคนี้ ผู้เขียนจะขอกล่าวถึงกระบวนการวิเคราะห์ Metagenome มีขั้นตอนคร่าวๆ ด้านล่างนี้ ส่วนละเอียดขอให้ผู้อ่านไปอ่านตามหัวข้อต่างๆในรายงาน

1. ในโปรเจคของการวิเคราะห์ Microbiome หนึ่งๆ (Bioproject) จะมีการนำเชื้อตัวอย่างไปเข้าเครื่อง sequencer ได้เป็น raw read data ของ Metagenome
2. ข้อมูล Read เหล่านี้จะถูกเก็บเป็นไฟล์ฟอร์แมตที่เรียกว่า FastQ หรือ FastA และข้อมูลนี้ถูกเรียกว่า Sequence Read Archive (SRA) ซึ่งการเข้าเครื่อง Sequencing แต่ละครั้งอาจจะได้ผลลัพธ์ไม่เหมือนกัน จึงมีการกำหนด ID ให้การ Sequence แต่ละครั้งว่า Sequence Read Run (SRR) ซึ่งใน Bioproject ใดๆ สามารถมี SRA ได้หลายชุด เนื่องจากนักวิจัยสามารถเก็บตัวอย่างหลายๆชุดมาวิเคราะห์ และ 1 SRA ก็มี SRR ได้หลายชุด
3. ข้อมูล SRA เหล่านี้สามารถนำไปวิเคราะห์ด้วยเครื่องมือหลากหลายเครื่องมือ เช่น QIIME2, MEGAN X แต่เครื่องมือที่ใช้กันอย่างแพร่หลายที่สุดคือ Blast เมื่อผ่านกระบวนการ Blast แล้วนักวิทยาศาสตร์จะสามารถคาดคะเนว่า ชุดข้อมูล SRA ใดๆ มีปริมาณสิ่งมีชีวิตแต่ละชนิดเท่าไหร่ (เป็นผลลัพธ์คร่าวๆตามค่า parameter ที่ปรับแต่งบน blast)
4. หลังจากกระบวกการ Blast นักวิทยาศาสตร์อาจจะอยากทราบข้อมูลเพิ่มเติมของ Bioproject หรือ SRA ใดๆ เช่น ข้อมูล Gene function, ข้อมูล Pathway นักวิทยาศาสตร์ก็จะใช้เครื่องมือเพิ่มเติมในการวิเคราะห์

1. สคริปสำหรับวิเคราะห์ข้อมูล Microbial Metagenome

* เพื่อความสะดวก ผู้เขียนจะขอเรียกแทนโปรเจคนี้ว่า ไปป์ไลน์

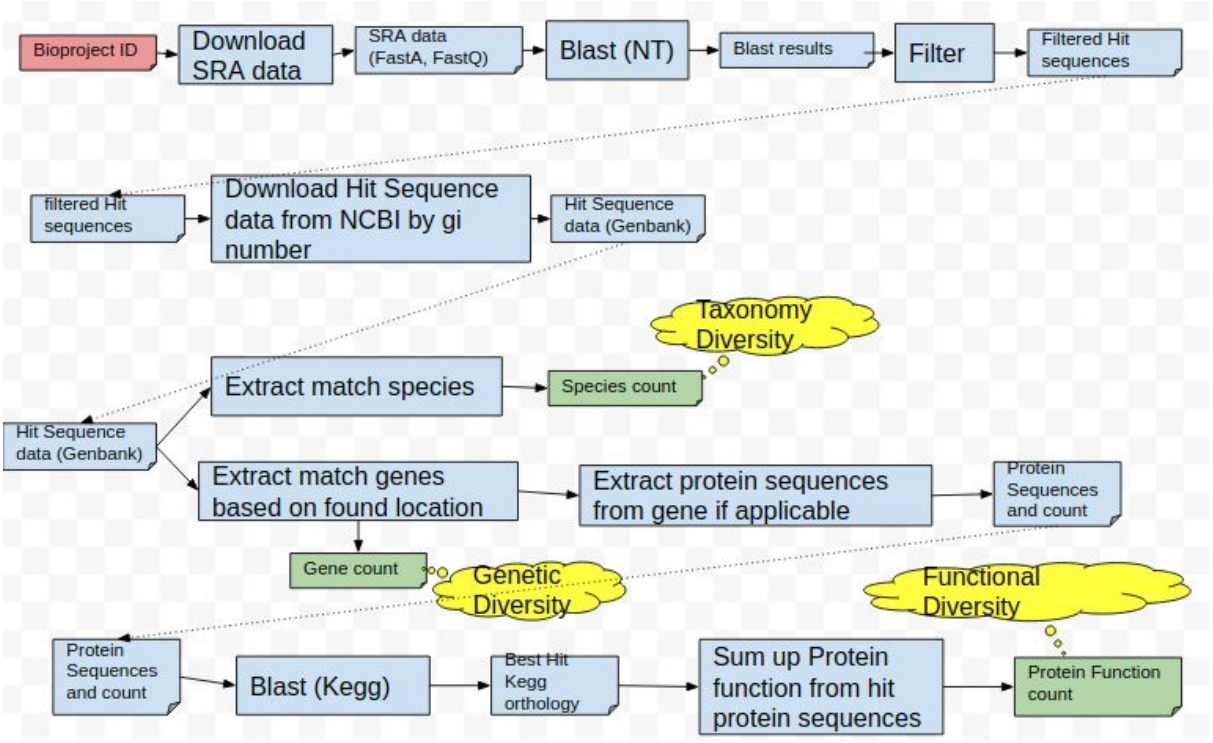
เป็นไปป์ไลน์ที่ใช้ในการวิเคราะห์ว่า ในแต่ละ Bioproject หรือ SRA มี Gene, Species และ โปรตีน อย่างละเท่าไหร่

ไฟล์อยู่ที่ : โฟลเดอร์ metagenomic_pipeline/ บน CD ที่แนบมากับรายงาน

Input : Bioproject id หรือ SRA id(s)

Output : ไฟล์ TSV 3 ไฟล์ ระบุจำนวน ยีน species และ Protein Function

กระบวนการทำงานของโปรเจคนี้



ภาพรวม workflow ของไปป์ไลน์

1. โปรเจคนี้เป็นไปป์ไลน์ที่เริ่มต้นที่ข้อมูล Bioproject หรือ SRA ซึ่งข้อมูลของแต่ละ Bioproject สามารถสร้างขึ้นมาได้จากวิธีการที่กล่าวด้านบน หรือ นักวิจัยสามารถไปดาวโหลดข้อมูล Bioproject ที่มีคนทำได้แล้ว ซึ่งฐานข้อมูลที่เก็บข้อมูล Bioproject นี้มีชื่อว่า NCBI ไปป์ไลน์นี้จะทำการดาวโหลด
2. กระบวนการการดาวโหลดข้อมูล Bioproject หรือ SRA มีดังนี้
    a. ผู้ใช้สามารถใช้ได้ทั้งเลข Bioproject และ SRA ซึ่งทางตัวโปรแกรมจะทำการวิเคราะห์ให้ว่า Input ที่เข้าไปนั้นเป็นข้อมูลชนิดใด
    b. ถ้าเป็นข้อมูล Bioproject ตัวโปรแกรมจะทำการดึงข้อมูลจาก NCBI หาว่ามี SRA อะไรเกี่ยวข้องกับ Bioproject นี้บ้างก่อน
    c. หลังจากได้เลข SRA ซึ่งอาจจะมีได้หลายชุด ทางโปรแกรมจะทำการดึงข้อมูลจาก NCBI อีกรอบเพื่อหาเลข SRR ล่าสุด จะก่อนดาวโหลดข้อมูลมาเป็น FastQ

d. ตัวโปรแกรมจะทำการเปลี่ยนข้อมูล FastQ เป็น FastA ให้อัตโนมัติ เพื่อรองรับการวิเคราะห์ในขั้นตอนต่อๆไป

3. ในการรันโปรแกรม Blast เนื่องด้วยข้อมูลมีขนาดใหญ่และใช้เวลานาน จึงควรใช้ High Performance Computer (HPC) ในการดำเนินการ ซึ่งทางไปป์ไลน์จะทำการ แปงไฟล์ FastA ของ SRA ที่ดาวโหลดมาให้อัตโนมัติ รวมถึงทำการ Blast แบบ Parallel ให้อัตโนมัติ แต่ผู้ใช้ต้องดำเนินการขั้นตอนนี้บน HPC ในกระบวนการ Blast ฐานข้อมูลที่ใช้ในการวิเคราะห์คือ NCBI NT ซึ่งเป็นฐานข้อมูลที่ใช้วิเคราะห์ Nucleotide ผู้ใช้สามารถปรับแต่งค่า parameter ต่างๆได้ที่ไฟล์ config.py

<qseqid>	<sseqid>	<sscinames>	<pident>	<length>	<mismatch>	<gapopen>	<qstart>	<qend>	<sstart>	<send>	<evalue>	<bitscore>
SRR2579826.361	gi|1095307724|gb|CP013019.1|	Clostridium pasteurianum	98.785	494	1	5	16	507	10189	9699	0.0	874
SRR2579826.361	gi|1095307724|gb|CP013019.1|	Clostridium pasteurianum	98.785	494	1	5	16	507	50715	50225	0.0	874
...	...	...	...									

ตัวอย่างผลลัพธ์จากการ Blast

4. หลังจากได้ผลลัพธ์จาก Blast ตัวโปรแกรมจะทำการรวมผลลัพธ์ให้เหลือไฟล์เดียว อีกทั้งจะทำการ Filter ข้อมูลเพียง 1 hit sequence ต่อ 1 query seqeunce โดยเลือกจาก Top Score ของ Blast ส่วน sequence ที่ไม่สามารถ hit จะถูกตัดออกจากการวิเคราะห์ไม่เอามาคำนวณเป็นผลลัพธ์
5. หลังจากได้ผลลัพธ์รวมของ Blast เราจะทราบว่า Query Sequences (ข้อมูล Sequnce จาก SRA บนคอลัมแรก) มีความคล้ายคลึงกับ Sequence ใดบนฐานข้อมูล (Subject / Hit Sequences บนคอลัมที่สอง) สิ่งที่โปรแกรมทำก็คือการนำ Subject ID เหล่านี้ไปวิเคราะห์ Taxonomy analysis (species count), gene analysis (gene count และ Protein function count) โดยมีรายละเอียดดังนี้
    a. โปรแกรมจะนำ Subject ID ทั้งหมดที่ได้จากการ Blast ไปค้นฐานข้อมูล NCBI และทำการดึงข้อมูล Taxonomy, gene, และ Protein sequence ของยีน ในกรณีที่ยีนนั้นเป็น Coding Sequence(CDS)
    b. Taxonomy analysis: โปรแกรมจะคัดเอาเฉพาะชื่อ Species จากข้อมูลที่ถูกดึงมา และทำการนับจำนวน Species แต่ละชนิด และให้ผลลัพธ์เป็น Text file
    c. Gene Analysis โปรแกรมจะคัดเอาเฉพาะชื่อยีนจากข้อมูลที่ถูกดึงมา และทำการนับจำนวนยีน และให้ผลลัพธ์เป็น Text file แสดง Bioproject ID, SRR(s), ชื่อยีนที่พบทั้งหมด และ จำนวนยีนแต่ละตัวที่พบ ในกรณีที่ผล BLAST hit ของ SRA sequence read พบ similarity ตรงเพียงบางส่วนของ contig, scaffold หรือโครโมโซม โปรแกรมจะตรวจสอบว่าตำแหน่งของ BLAST Hit location นั้นเป็นบริเวณของยีนหรือไม่ โปรแกรมจะนับเฉพาะผล BLAST hit ที่ตรงกับตำแหน่งที่เป็นยีนเท่านั้น ดังนั้นผลรวม read counts ของยีนทั้งหมดที่เจออาจจะมีจำนวนน้อยกว่าจำนวน read counts รวมทั้งหมดของ species ที่เจอ เพราะ sequence บางส่วนไม่มี BLAST hit ตรงกับบริเวณที่เป็นยีน
    d. ในกรณีที่ยีนสร้างโปรตีน ทางโปรแกรมจะทำการดึงข้อมูล Protein sequence ที่เป็น product ของยีนนั้นๆ และทำการเก็บข้อมูล Protein sequence ในฟอร์แมต fasta (ส่วนยีนที่ไม่สร้างโปรตีน เช่น R16S จะไม่ถูกรวมในกรณีนี้)
6. กระบวนการถัดไป เราอยากจะทำนายว่า Protein Sequences ที่พบนั้นมีหน้าที่อะไร (Protein Function) ผู้ใช้จะต้องนำ Protein Sequence ที่ไปป์ไลน์สร้างไว้ให้ ไปทำการ Blast กับ Kegg Prokaryotic Gene Database ที่เตรียมไว้ให้ แล้วเอาผลลัพธ์ที่เป็น Blast Format '6 qseqid stitle pident length mismatch gapopen evalue bitscore' มาดำเนินการต่อในไปป์ไลน์ สาเหตุที่ต้องตั้งค่านี้เพราะ Default Format จะไม่แสดงชื่อ Protein Function ตามที่เราต้องการ

<Function_id> <orthology_id> <function_name>	<count>
kpnk:BN49_1340 K11752 diaminohydroxyphosphoribosylaminopyrimidine deaminase / 5-amino-6-(5-phosphoribosylamino)uracil reductase [EC:3.5.4.26 1.1.1.193]	2

ตัวอย่างผลลัพธ์จากการ Blast กับ kegg database

7. ผลลัพธ์จากการ Blast กับ Kegg database นั้นจะบอกว่า protein sequence ที่พบนั้นมี Function อะไร ตัวไปป์ไลน์จะทำการนับจำนวน Protein Function ให้ และผลลัพธ์จะอยู่ในฟอร์แมต TSV

Technology หลักที่ใช้พัฒนา

1. Python 3 - เป็นภาษาโปรแกรมหลักที่ใช้เขียน
    a. Biopython - Python library ที่ใช้ในการเชื่อมต่อกับฐานข้อมูล NCBI
    b. Pandas - Python library สำหรับประมวลผล CSV
2. NCBI Blast 2.7.1+ - Basic Local Alignment Search Tool ใช้สำหรับค้นหา Sequences ที่ใกล้เคียง
3. SRA toolkit - ใช้สำหรับดาวโหลดข้อมูล SRA
4. High Performance Computer - ของ Biotec

ฐานข้อมูลที่ใช้

1. Kegg - ฐานข้อมูลเพื่อใช้ค้นหา Protein Function
2. NCBI Nt - ฐานข้อมูลเพื่อใช้ค้นหา nucleotide

ข้อเสนอแนะ

การวิเคราะห์ Protein Function

ปัจจุบันได้มีเครื่องมือหลายอย่างที่ช่วยอำนวยความสะดวกในการวิเคราะห์ Protein Function ในโปรเจคนี้เริ่มต้นด้วยการทำใช้ Interproscan แต่เนื่องจาก Interproscan นั้นใช้เวลานานในการวิเคราะห์ กล่าวคือ ไฟล์ ประมาณ protein sequence 1,000 Reads (ประมาณ 400 kB) อาจจะใช้เวลาประมาณ 3 วัน ซึ่งข้อมูล SRA อาจจะมี Protein Sequence มากกว่า 1,000,000 Reads โปรเจคนี้จึงได้ใช้ฐานข้อมูล KEGG เป็นหลักในการวิเคราะห์ protein function เพื่อจำแนกโปรตีนเป็น KEGG ortholog อันจะช่วยลดขนาดอินพุตสำหรับการวิเคราะห์ protein function ต่อไป

ตัวอย่างการใช้งานอย่างง่าย

สมมติว่าต้องการจะวิเคราะห์ Bioproject “PRJNA298936” ขั้นตอนง่ายๆดังนี้

1. ก็อปปี้โฟลเดอร์ metagenomic_pipeline/ ที่ประกอบด้วยไฟล์ 6 ไฟล์ไปบน Folder ใดๆ บน HPC และเข้าไปในโฟลเดอร์นั้น

2. ใช้คำสั่งด้านล่าง เพื่อดาวน์โหลดไฟล์และ วิเคราะห์ด้วย blast

```
python download_sra_data.py PRJNA298936 blast
```

3. หลังจาก Blast สำเร็จแล้ว ใช้คำสั่งด้านล่างเพื่อทำการสรุปผลวิเคราะห์จาก Blast

```
python postprocessing.py PRJNA298936
```

4. ผลลัพธ์อยู่ในโฟลเดอร์ metagenomic_pipeline/analysis/<BIOPROJECT_ID>/ ประกอบด้วย summary_gene_<BIOPROJECT_ID>, summary_species_<BIOPROJECT_ID> และ <BIOPROJECT_ID>_protein.fasta

* ในกรณีที่ต้องการนับจำนวน Protein Function ด้วยให้ทำดังนี้

5. นำเอา metagenomic_pipeline/analysis/<BIOPROJECT_ID>/<BIOPROJECT_ID>_protein.fasta ที่ได้จากข้อ 4 ไปทำการ Blast เทียบกับ Kegg database ตัวอย่างคำสั่ง Blast ดังนี้

```
blastp -outfmt '6 qseqid stitle pident length mismatch gapopen evalue bitscore' -query analysis/PRJNA298936/PRJNA298936_protein.fasta -db /gpfs/cluster/home/watchara/workspace/protein/keggdb/aaseq -max_target_seqs 1 -out out_test
```

6. เมื่อได้ผลลัพธ์จาก Blastp ให้ใช้คำสั่งด้านล่างเพื่อนับจำนวน Protein Function

```
python function_count.py out_test
```

7. ผลลัพธ์จะอยู่ที่ metagenomic_pipeline/analysis/<INPUT_NAME>_function_count.tsv ตัวอย่างของชื่อไฟล์ เช่น metagenomic_pipeline/analysis/out_test_function_count.tsv

<u>* กรณีที่ Input เป็น ไฟล์ FastA</u>

ในขั้นตอนที่ 2 ให้นำไฟล์ fastA ไปไว้ในโฟลเดอร์ SRA_data/fasta/ โดยตั้งชื่อว่า <SRR_ID>.fasta ให้ใช้คำสั่งดังตัวอย่างด้านล่าง

```
python blast_preprocess.py SRR2579826 blast
```

หรือถ้าต้องการโครงสร้างแบบ Bioproject ให้นำไฟล์ <SRR_ID>.fastA ของ SRR ที่มี ใส่ไปในโฟลเดอร์ SRA_data/fasta/ และให้ใช้คำสั่งเดิมในข้อที่ 2 เช่นด้านล่าง ตัวโปรแกรมจะไปตรวจสอบว่าไฟล์ SRR ดาวโหลดไว้แล้วหรือยัง ถ้าดาวโหลดแล้วจะไม่ดาวโหลดซ้ำ

```
python download_sra_data.py PRJNA298936 blast
```

* กรณีนี้ไม่รองรับไฟล์ FastQ เพราะว่ามีโอกาสที่ไฟล์ FastQ จะดาวโหลดไม่สมบูรณ์ ซึ่งตัวโปรแกรมจะไม่ทราบว่าดาวโหลดมาสมบูรณ์ไหม

Configuration File

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/config.py บน CD
คำอธิบาย: ใช้สำหรับปรับแต่งค่าต่างๆสำหรับไปป์ไลน์

```
EMAIL = อีเมล์ที่ส่งให้ NCBI
EMAIL = "xxx@gmail.com"

SPLIT_NUMBER = จำนวนคอร์ที่ใช้ในการ Blast
SPLIT_NUMBER = 200

BLASTDB_PATH = PATH ของ taxonomy database
BLASTDB_PATH = "/home/watchara/taxdb"

DB_PATH = PATH ของ Blast database ที่ใช้ Alignment
DB_PATH = "/home/somsaklik/BLASTDB/nt"

E_VALUE = ค่า E-Value cutoff BLAST
E_VALUE = 1e-05

MAX_TARGET_SEQ = จำนวนผลลัพธ์ที่แสดงต่อ 1 Query sequence
MAX_TARGET_SEQ = 1
```

ตัวอย่างผลลัพธ์

metagenomic_pipeline/analysis/PRJNA298936/summary_gene_PRJNA298936

Input bioproject : PRJNA298936	
Input SRRs : SRR2592316, SRR2592317, SRR2592314, SRR2579826, SRR2592315	
16S ribosomal RNA (ชื่อยีน)	75386 (จำนวน)
small subunit ribosomal RNA	1582
amplified using primers designed for 16S ribosomal RNA	13
Small Subunit Ribosomal RNA; ssuRNA; SSU ribosomal RNA	11
16S small subunit ribosomal RNA	9
bifunctional diaminohydroxyphosphoribosylaminopyrimidine deaminase/5-amino-6-(5-phosphoribosylamino)uracil reductase RibD	1
16S ribosomal RNA gene	1
internal transcribed spacer 2	1

16S ribosomal RNA (rrnC operon)	1
Original segmented set titles: E.coli T1 ribonuclease sensitive 16S rRNA region, segment 1.; E.coli T1 ribonuclease sensitive 16S rRNA region, segment 2.	1

ผลลัพธ์จาก Summary_gene รวมผลลัพธ์ Gene ทั้งหมดของทั้ง bioproject โดยเรียงลำดับจากการเจอมากไปหาน้อย

- ส่วนแรก คือ Bioproject ID
- ส่วนที่สอง คือ SRR ทั้งหมดที่ถูกรวม (หรือ SRR ทั้งหมดของ Bioproject นี้)
- ส่วนถัดไป คือ Gene count บอกชื่อและจำนวนยีนส์ที่พบใน Bioproject นี้

metagenomic_pipeline/analysis/PRJNA298936/summary_species_PRJNA298936

Input bioproject : PRJNA298936	
Input SRRs : SRR2592316, SRR2592317, SRR2592314, SRR2579826, SRR2592315	
Clostridium pasteurianum (ชื่อ Species)	41563 (จำนวน)
Klebsiella pneumoniae	15514
uncultured bacterium	14871
Klebsiella pneumoniae subsp. pneumoniae	2949
uncultured organism	1545
Escherichia coli	930
Salmonella enterica subsp. enterica serovar Albany	682
Klebsiella sp.	628
Klebsiella sp. UIWRF0518	536
Enterococcus faecalis	463
uncultured Klebsiella sp.	451
Shigella sp.	395
Klebsiella variicola	306
Escherichia sp.	277
Klebsiella sp. UIWRF0499	247
...	...

ผลลัพธ์จาก Summary_species รวมผลลัพธ์ Species ทั้งหมดของทั้ง bioproject โดยเรียงลำดับจากการเจอมากไปหาน้อย

- ส่วนแรก คือ Bioproject ID
- ส่วนที่สอง คือ SRR ทั้งหมดที่ถูกรวม (หรือ SRR ทั้งหมดของ Bioproject นี้)
- ส่วนถัดไป คือ Species count บอกชื่อและจำนวน Species พบใน Bioproject นี้

metagenomic_pipeline/analysis/PRJNA298936/PRJNA298936_protein.fasta

```
>AWY29003.1-c=1
MQDEMYMARALKLAARGRFTTHPNPNVGCVIVKDGEIVGEGFHYRAGEPHAEVHALRMAGEKARGATAYVT
LEPCSHHGRTPPCCDALIAAGVSRVVAAMQDPNPQVAGRGLYRLQQAGIEVSHGLMMNEAEALNKGFLKRM
RTGFPWVQLKLGASLDGRTAMASGESQWITSPQARRDVQRLRAQSHAILTSSATVLADDPALTVRWQELSAD
TQALYPEENLRQPLRVVIDSQNRVTPEHRIVQQAGETLFARLRADERQWPESARTLLVPEHNGHLDLVLLMMLL
GKQQINSVWVEAGATLAGALLQAGLVDELIVYIAPKLLGNAARGLCALPGLEELSQAPHFKFNEIRQVGPDVCLH
LTTA
```

ผลลัพธ์จาก Protein.fastA คือ Protein sequences ที่พบทั้งหมดใน Bioproject นี้

- ส่วนแรก (บันทัดที่มี “>”) บอก Protein ID - จำนวนที่พบ ดังตัวอย่าง Protein ID คือ AWY29003.1 และจำนวนที่พบ คือ 1 (c=1)
- ส่วนที่สอง คือ Protein Sequences (Amino acid sequences)

metagenomic_pipeline/analysis/out_test_function_count.tsv

<Function_id> <orthology_id> <function_name>	<count>
kpnk:BN49_1340 K11752 diaminohydroxyphosphoribosylaminopyrimidine deaminase / 5-amino-6-(5-phosphoribosylamino)uracil reductase [EC:3.5.4.26 1.1.1.193]	1

- ประกอบด้วย Kegg Ortholog number (K11752) และ จำนวนที่พบ (1)

คำอธิบายไปป์ไลน์โดยละเอียด

ระบบไปป์ไลน์นี้ถูกแยกออกเป็น 5 ส่วนหลักและ 1 Utilities เพื่อทำให้ง่ายต่อการเข้าใจและการใช้งาน ดังนี้

1. ส่วนดาวน์โหลดข้อมูล Sequence Read Archive (SRA)

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/download_sra_data.py บน CD

คำอธิบาย: ใช้สำหรับดาวน์โหลดข้อมูล SRA จากฐานข้อมูลของ NCBI ผ่านอินเตอร์เน็ต
Input: สามารถใช้ Input ได้หลายแบบดังนี้

- Bioproject ID เช่น PRJNA298936, "PRJNA298936"
- SRA ID เช่น SRS1435466, “SRS1435466”
- List ของ SRA ID เช่น SRS1107428, SRS1107429

* Input ที่กล่าวมาให้ใส่ไปใน Command Line เป็น Command line parameter

ตัวอย่าง Input ทาง Command Line

- “python download_sra_data.py PRJNA298936”
- “python download_sra_data.py SRS1107428, SRS1107429, SRS1107427, SRS1107426, SRS1107419”

*หมายเหตุ สามารถใส่ keyword “blast” ร่วมไปใน Input เพื่อให้เรียกใช้งานส่วนที่ 2 (Blast) อัตโนมัติ โดย keyword blast จะอยู่ก่อน หรือหลัง Bioproject/SRA ID ก็ได้ เช่น

- “python download_sra_data.py PRJNA298936 blast”
- “python download_sra_data.py blast SRS1107428, SRS1107429, SRS1107427, SRS1107426, SRS1107419”

Output:

- ไฟล์ที่ดาวน์โหลด จะอยู่ใน Format ของ FastQ และ FastA (FastA ได้จากการแปลงข้อมูลจาก FastQ โดยตัวระบบ) จะอยู่ในโฟลเดอร์ metagenomic_pipeline/SRA_data/ ซึ่งประกอบด้วย
    - FastA = ไฟล์ใน Format fastA
    - FastQ = ไฟล์ใน Format fastQ
        - Fastq.log = เก็บเลข SRR ที่ถูกดาวน์โหลดเสร็จสมบูรณ์ไว้
    - download_<date>.log = รายละเอียดของข้อมูลที่ดาวน์โหลด

2. ส่วนการวิเคราะห์ข้อมูลด้วย Basic Local Alignment Search Tool (Blast)

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/blast_preprocess.py บน CD

คำอธิบาย: ใช้สำหรับ แตก fastA เป็นหลายๆไฟล์ย่อย เพื่อใช้ในการ Blast

Input:

- SRA ID เช่น SRS1435466, “SRS1435466”
- List ของ SRA ID เช่น SRS1107428, SRS1107429

* Input ที่กล่าวมาให้ใส่ไปใน Command Line เป็น Command line parameter

ตัวอย่าง Input ทาง Command Line

- “python blast_preprocess.py SRR2579826”
- “python postprocessing.py SRR2579826, SRR2592315, SRR2592314, SRR2592316, SRR2592317”

*หมายเหตุ สามารถใส่ keyword “blast” ร่วมไปใน Input เพื่อให้เรียกใช้งาน Blast อัตโนมัติ โดย keyword blast จะอยู่ก่อน หรือหลัง Bioproject/SRA ID ก็ได้ เช่น

- “python download_sra_data.py SRR2579826 blast”
- “python download_sra_data.py blast SRR2579826, SRR2592315, SRR2592314, SRR2592316, SRR2592317”

*หมายเหตุ 2 ควรเรียกใช้งานส่วนนี้บน HPC

Output:

- ไฟล์ย่อย (ไฟล์ SRR จะถูกแบ่งเป็น 200ไฟล์ย่อย ในฟอร์แมต FastA เพื่อใช้ในการ Parallel) ที่ถูกแยกจากไฟล์หลัก (Input) ในโฟลเดอร์ metagenomic_pipeline/blast/splitted_<SRR_ID>/
- metagenomic_pipeline/blast/splitted_<SRR_ID>/run_blast.sh คือ script ที่ใช้รัน Blast บน HPC
- * กรณีที่เรียกใช้ Blast อัตโนมัติ จะได้ผลลัพธ์จาก blast output ในโฟลเดอร์ของไฟล์ย่อยด้วยที่ metagenomic_pipeline/blast/splitted_<SRR_ID>/

3. ส่วนวิเคราะห์ผลลัพธ์จาก Blast

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/postprocessing.py บน CD

คำอธิบาย: ใช้สำหรับนับจำนวน Species, Gene และแสดง Protein sequences ทั้งหมด ที่ได้จาก SRR

Input:

- Bioproject ID เช่น PRJNA298936, "PRJNA298936"
- SRA ID เช่น SRS1435466, “SRS1435466”
- List ของ SRA ID เช่น SRS1107428, SRS1107429

ตัวอย่าง Input ทาง Command Line

- “python download_sra_data.py PRJNA298936”
- “python download_sra_data.py SRS1107428, SRS1107429, SRS1107427, SRS1107426, SRS1107419”

Output:

- ไฟล์ที่บอกว่า แต่ละ Species, Gene มีจำนวนเท่าไร
- ข้อมูล Output จะอยู่ในโฟลเดอร์ metagenomic_pipeline/analysis/<SRR_ID>/ ซึ่งประกอบด้วย
    - ข้อมูลจำนวน gene ในไฟล์ <SRR_ID>_gene.tsv
    - ข้อมูลจำนวน species ในไฟล์ <SRR_ID>_species.tsv
    - ข้อมูลรายละเอียดของผลลัพธ์ทั้งหมดใน <SRR_ID>.csv
    - Protein sequences ทั้งหมดที่เจอใน โฟลเดอร์ protein/
- นอกจากนี้ยังมี metagenomic_pipeline/cache/ ซึ่งเก็บข้อมูลของ species ต่างๆ ที่ดาวน์โหลดจาก NCBI ในกรณีที่ cache/ ต่างใหญ่เกินไป จะทำให้การทำงานส่วนนี้ช้าลง จึงควรลบทิ้งเป็นระยะๆ
- * โปรแกรมในส่วนนี้จะเรียกใช้ โปรแกรมในส่วนที่ 4 (metagenomic_pipeline/combine_result.py) อัตโนมัติ

4. ส่วนรวมผลลัพธ์เพื่อนำไปใช้งานจริง

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/combine_result.py บน CD

คำอธิบาย: ใช้สำหรับรวมผลลัพธ์การนับจำนวน Species, Gene และ รวม protein sequences ของทั้ง Bioproject ซึ่งได้จาก postprocessing.py กล่าวคือ ในกรณีที่ต้องการรู้จำนวนผลลัพธ์ของทั้ง Bioproject แต่ว่าโปรแกรมจากข้อ1-3 สามารถทำทีละ 1 SRA ขั้นตอนนี้จะเป็นการรวมผลลัพธ์กลับไปเป็น Bioproject ผลลัพธ์ทั้งหมดจากขั้นตอนที่ผ่านมา จะถูกย้ายไปรวมกันที่โฟลเดอร์ metagenomic_pipeline/analysis/<BIOPRJECT_ID>/

Input:

- List ของ SRA ID เช่น SRS1107428, SRS1107429

* Input ที่กล่าวมาให้ใส่ไปใน Command Line เป็น Command line parameter ดังนี้

- “python combine_result.py SRR2579826, SRR2592315, SRR2592314, SRR2592316, SRR2592317”

Output:

- ไฟล์ที่บอกว่า ผลลัพธ์รวมทั้งหมดแต่ละ Species, Gene มีจำนวนเท่าไหร่
- ข้อมูล Output จะถูกใส่ไปในโฟร์เดอร์ metagenomic_pipeline/analysis/<BIOPRJECT_ID>/ ซึ่งประกอบด้วย
    - ข้อมูลจำนวน gene ในไฟล์ summary_gene_<date-time>
    - ข้อมูลจำนวน species ในไฟล์ summary_species_<date-time>
    - ข้อมูล Protein sequences ทั้งหมดใน Bioproject ในไฟล์ <BIOPROJECT_ID>_protein.fasta

5. ส่วนนับจำนวน Protein Functions

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/function_count.py บน CD
คำอธิบาย: ใช้สำหรับนับจำนวน Protein function จากผลลัพธ์ที่ได้จากการเอา Protein Sequences ไป Blast กับ Kegg database

- ตัวอย่างคำสั่งที่ใช้ในการ Blast

```
blastp -outfmt '6 qseqid stitle pident length mismatch gapopen evalue bitscore' -query analysis/PRJNA298936/PRJNA298936_protein.fasta -db /gpfs/cluster/home/watchara/workspace/protein/keggdb/aaseq -max_target_seqs 1 -out out_test
```

- Outfmt : format ของ Blast output ให้ใช้ '6 qseqid stitle pident length mismatch gapopen evalue bitscore' เท่านั้น เพราะค่า Default จะไม่แสดงชื่อ Protein Function
- Query : FastA file ที่จะใช้ค้นหา
- Db : database ที่ใช้ ก็คือ Kegg database
- Max_target_seqs : จำนวน hit sequences ที่จะแสดงผลลัพธ์ ต่อ 1 query sequences
- Out : ชื่อไฟล์ output

Input:

- ชื่อไฟล์ที่จะใช้นับจำนวน ไฟล์นี้คือผลลัพธ์ที่ได้จากการนำ Protein Seqeunces ไป Blast กับ Kegg Database
- ตัวอย่างการใช้งาน สมมติผลจาก Blast อยู่ในไฟล์ out_test สามารถ Run โปรแกรม “python function_count.py out_test”

Output:

- ไฟล์ tsv ที่บอกผลรวมของ Protein Function แต่ละแบบมีจำนวนอย่างละเท่าไหร่
- ข้อมูล Output จะถูกใส่ไปในโฟลเดอร์ metagenomic_pipeline/analysis/<INPUT_NAME>_function_count.tsv

6. Utility functions

ชื่อไฟล์ที่เกี่ยวข้อง: metagenomic_pipeline/utils.py บน CD
คำอธิบาย: ใช้สำหรับเก็บ function ที่ถูกเรียกจากหลายๆไฟล์

เปรียบเทียบประสิทธิภาพการทำงานของไปป์ไลน์

ในที่นี้ผู้เขียนจะขอใช้ Bioproject ID = PRJNA298936 ซึ่งประกอบด้วย SRR2579826 (14381 sequences, 8.5MB), SRR2592314 (18119 sequences, 10.9MB), SRR2592315 (14381 sequences, 8.5MB), SRR2592316 (15803 sequences, 9.4MB), SRR2592317 (22076 sequences, 13.3MB), มาเปรียบเทียบระหว่างผลลัพธ์จากไปป์ไลน์ที่สร้างขึ้นมาเอง กับผลลัพธ์ที่ได้จาก NCBI และ Qiime 2 classifier เพื่อให้ทราบถึงประสิทธิภาพการทำงานของไปป์ไลน์ ในส่วนของ Qiime 2 ผู้เขียนได้ใช้ Pretrained Model ชื่อ Naive Bayes ที่เทรนบนฐานข้อมูล Greengene

โดยจะขอแยกการเปรียบเทียบออกเป็นหัวข้อดังนี้

1. จำนวน Sequence

จำนวนผลรวมของ Sequence ทั้งของ Bioproject PRJNA298936 (ผลรวม Sequence ของ SRR ที่กล่าวไปข้างต้น) ที่ถูกดาวน์โหลดมาจาก NCBI มีจำนวนสาย nucleotide ทั้งหมด 84,760 Reads ผลลัพธ์ที่ได้จากไปป์ไลน์มีทั้งหมด 83,044 Reads จากNCBIมีทั้งหมด 99,065 Reads และจาก Qiime 2 มีทั้งหมด 84,760 Reads

ข้อมูลจริง	ไปป์ไลน์	NCBI	Qiime 2
84,760	84,712	99,068	84,760

ข้อมูลจากไปป์ไลน์มีน้อยกว่าความเป็นจริงเพราะในขั้นตอนการ Blast สาย nucleotide บาง read ไม่พบ similar hits ที่มีนัยสำคัญ (E-Value cutoff = 10^{-5}) จึงไม่ถูกนับ ผลลัพธ์จาก Qiime 2 จะไม่ได้ตัดออก แต่จะแสดงผลลัพธ์ว่าไม่สามารถทำนายได้แทน ส่วนผลลัพธ์จาก NCBI มีเกินจากความเป็นจริง ซึ่งอาจจะเกิดจากความผิดพลาดจากการดาวโหลดข้อมูล SRA, ดาวโหลดข้อมูล Species, หรือ วิธีการวิเคราะห์จาก NCBI ยังไม่แม่นยำเพียงพอ

2. ความสามารถในการจำแนก Taxonomy

Rank	**Pipeline**	**NCBI**	**QIIME 2**
Superkingdom	83,044	99,065	84,760
Phylum	67,993	80,551	79,083
Class	67,842	65,042	78,226
Order	67,755	64,684	78,116
Family	67,713	62,867	77,405
Genus	67,494	42,689	57,253
Species	59,948	402	31,035

ตารางแสดงจำนวนข้อมูลที่พบในแต่ละ Rank

จากตาราง ไปป์ไลน์ที่พัฒนาขึ้นในโปรเจ็กต์นี้, ไปป์ไลน์ของ NCBI, และ QIIME 2 ให้ผลการจำแนก Taxonomy ของแต่ละระดับออกมาแตกต่างกัน ผลการจำแนกในระดับ higher taxa (ตั้งแต่ระดับ genus ขึ้นไป) ของไปป์ไลน์ทั้งสามมีความใกล้เคียงกัน แต่ในระดับสปีชีส์มีความแตกต่างกัน ไปป์ไลน์ที่สร้างขึ้นมานั้นข้อมูลส่วนใหญ่สามารถจำแนกไปได้ถึงระดับ Species ได้มากกว่า NCBI และ QIIME 2

3. ความเร็วในการคำนวณผลลัพธ์

ไปป์ไลน์	NCBI	Qiime 2

7.5 ซ.ม.	n/a	2.5 ซ.ม.

ตารางแสดงความเร็วในการคำนวณผลลัพธ์

จะเห็นว่าไปป์ไลน์ที่สร้างขึ้นใช้เวลาเยอะกว่าการใช้ Qiime 2 จำแนกผลลัพธ์มาก เพราะ ไปป์ไลน์ที่สร้างขึ้นมามีใช้การ Blast ซึ่งใช้การ search database และดาวโหลดข้อมูลจาก NCBI ที่ใช้เวลาค่อนข้างสูง เมื่อเทียบกับการใช้ Machine Learning ของ Qiime 2 ซึ่งใช้การคำนวณ ในกรณีที่ข้อมูลมีปริมาณมากๆ ไปป์ไลน์ที่สร้างขึ้นอาจจะใช้เวลาในการดำเนินการสูงมาก (เช่น อาจจะเป็นเดือน) ในกรณีนี้ Qiime 2 อาจจะมีวิธีการที่เหมาะสมมากกว่า

* ในขั้นตอน blast อาจจะลดเวลาลงมาได้อีกโดยการใช้ database เฉพาะเจาะจงกับข้อมูลที่ต้องการค้นหา เช่น เปลี่ยนไปใช้ฐานข้อมูลที่มีแต่ข้อมูล microbial taxa เฉพาะ

4. ความแม่นยำของการจำแนกข้อมูล

การเปรียบเทียบผลการจำแนกจาก 3 โปรแกรมด้วยข้อมูลตัวอย่าง Bioproject ID = PRJNA298936 ในแต่ละระดับ taxonomic rank ได้ผลดังนี้

Superkingdom

	Pipeline	**NCBI**	**QIIME 2**
Bacteria	83,043	99,064	84,760
Eukaryota	1	1	0

Phylum

	Pipeline	**NCBI**	**QIIME 2**
Firmicutes	42,952	50,441	44,714
Proteobacteria	25,022	30,110	33,763
OD1			595
Actinobacteria	12		5
Bacteroidetes	3		4
Cyanobacteria	2		
Nitrospirae	2		1
Chloroflexi			1

Class

	Pipeline	NCBI	QIIME 2
Clostridia	42,152	44,478	43,764
Gammaproteobacteria	24,968	20,487	33,642
Bacilli	663	63	796
Betaproteobacteria	21	2	17
Alphaproteobacteria	19		23
Actinobacteria	12		5
Erysipelotrichia		9	
Bacteroidia			4
Sphingobacteriia	3		
Nitrospinia	2		
Tissierellia	2	2	
Negativicutes	0	1	
Nitrospira			1
Chloroflexi			1

Order

	Pipeline	NCBI	QIIME 2
Clostridiales	42,151	44,478	43,760
Enterobacterales	24,826	20,158	33,517
Lactobacillales	634	31	713
Xanthomonadales	30		46
Bacillales	29		13
Burkholderiales	20		17
Pseudomonadales	19		23
Rhizobiales	18		15
Erysipelotrichales		9	
Streptomycetales	6		

Actinomycetales	5		5
Bacteroidales			4
Cellvibrionales	3		
Sphingobacteriales	3		
Tissierellales	2	2	
Alteromonadales	2		
Nitrospinales	2		
Oscillatoriales	2		
Sphingomonadales	1		1
Methylophilales		2	

จากตารางทั้งหมด จะเห็นว่าผลลัพธ์จากไปป์ไลน์ กับ QIIME 2 จะค่อนข้างเหมือนกัน มากกว่าผลลัพธ์ที่ได้จาก NCBI

5. การจำแนก Gene และ Protein Function

นอกจาก Taxonomy classification แล้ว ไปป์ไลน์ที่สร้างขึ้นยังสามารถจำแนกยีนและ Protein Function ได้อีกด้วย แต่ NCBI และ QIIME 2 ไม่ได้รับรองฟังก์ชั่นนี้

2. ฐานข้อมูล Kegg Prokaryotic proteins

ฐานข้อมูล Kegg เป็นฐานข้อมูลที่เก็บ Ortholog ของโปรตีน กล่าวคือ โปรตีนที่มีฟังชั่นเหมือนกันจะถูกเก็บร่วมกันนอกจากนี้ฐานข้อมูล Kegg ยังเก็บข้อมูล Protein function ด้วย เพราะเหตุนี้โปรเจคนี้จึงเลือกใช้ฐานข้อมูล Kegg Prokaryotic Gene ในการหา Protein Function

โปรเจคนี้ได้ทำการสร้างโปรแกรมสำหรับดึงฐานข้อมูล Kegg Prokaryote Gene จากเซิฟเวอร์ของ Kegg และยังทำการติดตั้งฐานข้อมูลนี้ลงบนเซิฟเวอร์ HPC ของ Biotec อีกด้วย ดังข้อมูลที่แสดงด้านล่าง

ไฟล์อยู่ที่ : โฟลเดอร์ keggdb/ บน CD ที่แนบมากับรายงาน

ฐานข้อมูลนี้ คือ ฐานข้อมูล Prokaryote จากฐานข้อมูล Kegg Genes (https://www.kegg.jp/kegg/genes.html) ซึ่งถูกดาวน์โหลดผ่าน API https://www.kegg.jp/kegg/rest/keggapi.html ฐานข้อมูลนี้ รวบรวม Protein sequences (Amino Acid Sequences) ของ ยีนจาก Prokaryotes เพื่อใช้ในการ Blast ณ ปัจจุบัน มีจำนวนทั้งหมด 18,261,806 protein sequences และ 5,233 organisms ประกอบด้วยไฟล์ 3 ชนิด ดังนี้

1. **Amino acid sequences**

* ไฟล์นี้ถูกเก็บบน HPC ของ BIOTEC (ไม่มีบน CD) ที่ PATH sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy%20project/keggdb/download/aaseq/*.fasta

จะมีไฟล์ FastAทั้งหมด 5,233 ไฟล์ ตามจำนวน Species ของ Prokaryote ที่ถูกเก็บไว้ใน Kegg Database

Format ของข้อมูลเป็นดังนี้

> <ORGANISM_ABBREVIATION>:<FUNCTION_ID> <ORTHOLOGY_ID> <KO> | (RefSeq) <FUNCTION_ABBREVIATION>; <FUNCTION>
<PROTEIN SEQUENCE>

```
>eco:b0001 K08278 thr operon leader peptide | (RefSeq) thrL; thr operon leader peptide (A)
MKRISTTITTTITITTGNGAG
>eco:b0002 K12524 bifunctional aspartokinase / homoserine dehydrogenase 1 [EC:2.7.2.4 1.1.1.3]
| (RefSeq) thrA; fused aspartate kinase/homoserine dehydrogenase 1 (A)
MRVLKFGGTSVANAERFLRVADILESNARQGQVATVLSAPAKITNHLVAMIEKTISGQDA
LPNISDAERIFAELLTGLAAAQPGFPLAQLKTFVDQEFAQIKHVLHGISLLGQCPDSINA
ALICRGEKMSIAIMAGVLEARGHNVTVIDPVEKLLAVGHYLESTVDIAESTRRIAASRIP
ADHMVLMAGFTAGNEKGELVVLGRNGSDYSAAVLAACLRADCCEIWTDVDGVYTCDPRQV
PDARLLKSMSYQEAMELSYFGAKVLHPRTITPIAQFQIPCLIKNTGNPQAPGTLIGASRD
EDELPVKGISNLNNMAMFSVSGPGMKGMVGMAARVFAAMSRARISVVLITQSSSEYSISF
CVPQSDCVRAERAMQEEFYLELKEGLLEPLAVTERLAIISVVGDGMRTLRGISAKFFAAL
ARANINIVAIAQGSSERSISVVVNNDDATTGVRVTHQMLFNTDQVIEVFVIGVGGVGGAL
LEQLKRQQSWLKNKHIDLRVCGVANSKALLTNVHGLNLENWQEELAQAKEPFNLGRLIRL
VKEYHLLNPVIVDCTSSQAVADQYADFLREGFHVVTPNKKANTSSMDYYHQLRYAAEKSR
RKFLYDTNVGAGLPVIENLQNLLNAGDELMKFSGILSGSLSYIFGKLDEGMSFSEATTLA
REMGYTEPDPRDDLSGMDVARKLLILARETGRELELADIEIEPVLPAEFNAEGDVAAFMA
NLSQLDDLFAARVAKARDEGKVLRYVGNIDEDGVCRVKIAEVDGNDPLFKVKNGENALAF
YSHYYQPLPLVLRGYGAGNDVTAAGVFADLLRTLSWKLGV
>eco:b0003 K00872 homoserine kinase [EC:2.7.1.39] | (RefSeq) thrB; homoserine kinase (A)
MVKVYAPASSANMSVGFDVLGAAVTPVDGALLGDVVTVEAAETFSLNNLGRFADKLPSEP
RENIVYQCWERFCQELGKQIPVAMTLEKNMPIGSGLGSSACSVVAALMAMNEHCGKPLND
TRLLALMGELEGRISGSIHYDNVAPCFLGGMQLMIEENDIISQQVPGFDEWLWVLAYPGI
KVSTAEARAILPAQYRRQDCIAHGRHLAGFIHACYSRQPELAAKLMKDVIAEPYRERLLP
GFRQARQAVAEIGAVASGISGSGPTLFALCDKPETAQRVADWLGKNYLQNQEGFVHICRL
DTAGARVLEN
>eco:b0004 K01733 threonine synthase [EC:4.2.3.1] | (RefSeq) thrC; threonine synthase (A)
MKLYNLKDHNEQVSFAQAVTQGLGKNQGLFFPHDLPEFSLTEIDEMLKLDFVTRSAKILS
AFIGDEIPQEILEERVRAAFAFPAPVANVESDVGCLELFHGPTLAFKDFGGRFMAQMLTH
IAGDKPVTILTATSGDTGAAVAHAFYGLPNVKVVILYPRGKISPLQEKLFCTLGGNIETV
AIDGDFDACQALVKQAFDDEELKVALGLNSANSINISRLLAQICYYFEAVAQLPQETRNQ
LVVSVPSGNFGDLTAGLLAKSLGLPVKRFIAATNVNDTVPRFLHDGQWSPKATQATLSNA
MDVSQPNNWPRVEELFRRKIWQLKELGYAAVDDETTQQTMRELKELGYTSEPHAAVAYRA
LRDQLNPGEYGLFLGTAHPAKFKESVEAILGETLDLPKELAERADLPLLSHNLPADFAAL
RKLMMNHQ
>eco:b0005 no KO assigned | (RefSeq) yaaX; DUF2502 domain-containing protein YaaX (A)
MKKMQSIVLALSLVLVAPMAAQAAEITLVPSVKLQIGDRDNRGYYWDGGHWRDHGWWKQH
YEWRGNRWHLHGPPPPPRHHKKAPHDHHGGHGPGKHHR
...
```

2. **Gene Function List**

* ไฟล์นี้ถูกเก็บบน HPC ของ BIOTEC (ไม่มีบน CD) ที่ PATH sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy%20project/keggdb/download/kegg_protein_fn.txt

ไฟล์นี้ระบุ Gene ID และ ชื่อของ Function ที่ทำ โดยไฟล์นี้เป็นการสรุปรายการจากข้อมูลในส่วนแรก

function_id	function_name
eco:b0001	thrL; thr operon leader peptide
eco:b0002	thrA; fused aspartate kinase/homoserine dehydrogenase 1
eco:b0003	thrB; homoserine kinase
eco:b0004	thrC; threonine synthase
eco:b0005	yaaX; DUF2502 domain-containing protein YaaX
eco:b0006	yaaA; peroxide stress resistance protein YaaA
eco:b0007	yaaJ; putative transporter YaaJ
eco:b0008	talB; transaldolase B
eco:b0009	mog; molybdopterin adenylyltransferase
eco:b0010	satP; acetate/succinate:H(+) symporter
...	...

3. **ชื่อ และ Taxonomy** ของสิ่งมีชีวิตต่างๆ

* ไฟล์นี้ถูกเก็บบน HPC ของ BIOTEC (ไม่มีบน CD) ที่ PATH sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy%20project/keggdb/download/organism.tsv

Format ของข้อมูลเป็นดังนี้

<ORG_ID>	<ORG_ABB>	<ORGANISM_NAME>	<TAXONOMY>
T00007	eco	Escherichia coli K-12 MG1655	Prokaryotes;Bacteria;Gammaproteobacteria - Enterobacteria;Escherichia
T00068	ecj	Escherichia coli K-12 W3110	Prokaryotes;Bacteria;Gammaproteobacteria - Enterobacteria;Escherichia
T00666	ecd	Escherichia coli K-12 DH10B	Prokaryotes;Bacteria;Gammaproteobacteria - Enterobacteria;Escherichia
T00913	ebw	Escherichia coli BW2952	Prokaryotes;Bacteria;Gammaproteobacteria - Enterobacteria;Escherichia
T02541	ecok	Escherichia coli K-12 MDS42	Prokaryotes;Bacteria;Gammaproteobacteria - Enterobacteria;Escherichia
...	...	...	...

ไฟล์ที่เกี่ยวข้อง

1. keggdb/download_kegg.py script ที่ใช้ในการดาวโหลดข้อมูล kegg

2. keggdb/makedb.sh shell script ใช้ในการเปลี่ยนฐานข้อมูลนี้ ไปเป็น Blast database สามารถรันโดยการ ./makedb.sh บน Terminal
3. keggdb/keggdb/ เก็บ Blast database ที่ถูกสร้างแล้ว

* ไฟล์นี้ถูกเก็บบน HPC ของ BIOTEC (ไม่มีบน CD) ที่ PATH
sftp://watchara@203.185.133.166/gpfs/cluster/home/watchara/aiy%20project/keggdb/keggdb/

วิธีการดาวโหลด Kegg

รันคำสั่ง

```
python download_kegg.py
```

อธิบายวิธีการดาวโหลดโดยละเอียด

1. ข้อมูลของ Kegg API https://www.kegg.jp/kegg/rest/keggapi.html
2. ตรวจสอบรายชื่อ Organism ทั้งหมดที่ http://rest.kegg.jp/list/organism
3. นำ Organism ID คอลัมน์แรกของทุกแถวที่เป็น prokaryote) ของ Prokaryote ไปหาโปรตีน Function ที่ http://rest.kegg.jp/list/<organism id> เช่น http://rest.kegg.jp/list/T00007
4. นำ Protein Function ID (แถวแรกสุด) ไปค้นหา Amino acid sequences จาก http://rest.kegg.jp/get/<Protein Function ID(s)>/aaseq เช่น http://rest.kegg.jp/get/eco:b0001/aaseq
- สามารถใช้เครื่องหมาย + ขั้นเพื่อค้นหาได้ เช่น http://rest.kegg.jp/get/eco:b0001+eco:b0002+eco:b0003/aaseq
- Kegg อนุญาติให้ค้นหา Amino acid sequences ได้อย่างมากทีละ 10 ตัว ซึ่งถ้า query เกิน 10 ตัว ทางเซิฟเวอร์ของ Kegg จะส่งข้อมูลคืนกลับมาแค่ 10 ตัวแรก

5. นำข้อมูลแต่ละส่วนมาเก็บไว้ใน File

3. การทดลองวิเคราะห์ Microbiome ด้วย Network Visualization

ทฤษฎีกราฟเป็นสาขาวิชาที่ค่อนข้างกว้าง โดยส่วนที่สามารถประยุกต์ใช้กับงาน Microbiome ได้ เช่น การ Visualization, gene-gene / protein-protein / species-species interaction Link prediction, Node/edge function prediction, Community prediction / Clustering ทั้งนี้การจะเลือกใช้วิธีการวิเคราะห์ข้อมูลให้มีประสิทธิภาพจะต้องเข้าใจในลักษณะของข้อมูล และทางผู้เขียนได้หยิบ Network Visualization มาสร้างเป็นเว็บไซต์เพื่อเป็นการเริ่มต้นของการวิเคราะห์ข้อมูล Microbiome อีกด้วย

Network visualization เป็นวิธีในการวิเคราะห์ Species ของ Microbiome ที่มีประสิทธิภาพวิธีหนึ่ง เพราะความสัมพันธ์ของข้อมูลต่างๆ ไม่ว่าจะเป็น Species ต่อ Species หรือ Species ต่อ สิ่งแวดล้อม มีความซับซ้อน และยากที่จะดูด้วยตาเปล่า อีกทั้งการแสดงเป็นกราฟ สามารถหาแนวโน้มต่างๆของข้อมูลได้อีกด้วย ทางผู้เขียนจึงได้สร้างเว็บไซต์สำหรับแสดงข้อมูล Species

ผู้เขียนได้ศึกษาหาความเป็นไปได้ของการทำ Network Visualization ด้วยข้อมูล Microbiome ตัวอย่าง และสาธิตขั้นตอนและผลของการศึกษา ดังที่แสดงด้านล่าง

การเลือกใช้ Library

ในการทำ Network analysis มี Library ให้เลือกใช้ โดยมีทั้ง Frontend และ Backend โดย library ที่คนนิยมใช้มีดังนี้

1. Cytoscape.js คือ Javascript library สำหรับทำ Network Analysis ใช้ง่ายและมีความยืดหยุ่นสูง เหมาะสำหรับทำเว็บ ข้อเสียคือ ไม่รองรับ Layout แบบ Force Atlas 2 ที่ Gephi ใช้ ซึ่งเป็น Layout ที่ได้รับความนิยมสูงที่สุด แต่ก็ยังมี Layout แบบ Cose ที่พอใช้งานแทนกันได้ ตามที่แสดงบนเว็บไซต์
2. Sigma.js เป็น Javascript Library ที่ออกแบบมาเพื่อใช้กับ Gephi ปัญหาคือ ใช้งานยาก ซับซ้อน Document ไม่ดี และ ยืดหยุ่นไม่สูง
3. Vis.js / D3.js เป็น Javascript library สำหรับทำ General Visualization ซึ่งมีความยืดหยุ่นสูงมาก แต่ใช้งานยากมากเช่นกัน ไม่รองกับทฤษฎี Network Analysis เท่าไหร่นัก
4. NetworkX คือ Python library ที่ใช้ทำงานกับ Network Analysis ข้อดีคือฟังชั่นเยอะ ข้อเสียคือ Visualization ช้ามาก และไม่เหมาะกับงานของโปรเจคนี้ที่จะทำเว็บ
5. Gephi คือ Offline application ที่ได้รับความนิยมในการวิเคราะห์ Network Analysis ข้อดีของ Gephi คือ สามารถรองรับข้อมูลขนาดใหญ่ได้ และใช้งานแบบ Drag & Drop

ตารางเปรียบเทียบคุณสมบัติในด้านต่างๆ ของ Network analysis libraries

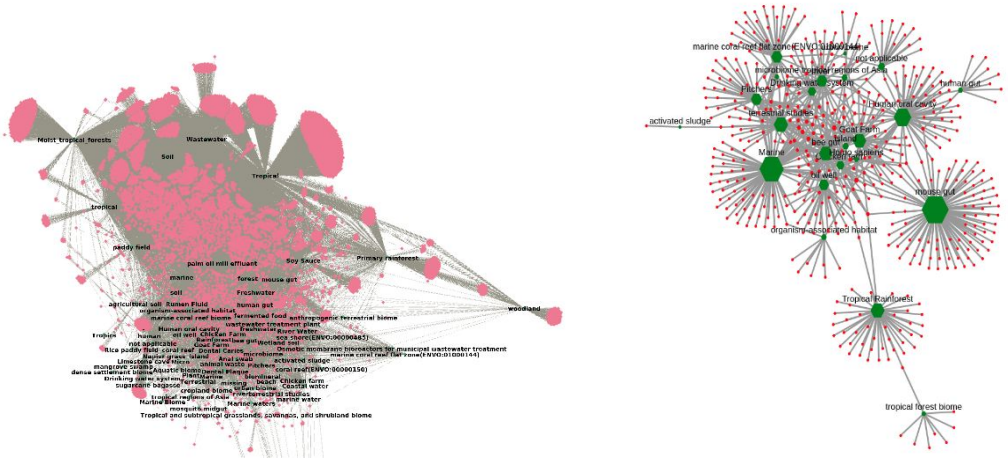
	Gephi	**NetworkX**	**CytoscapeJS**	**SigmaJS**	**VisJS/D3JS**
Programming Language	n/a (drag&drop)	Python	Javascript	Javascript	Javascript
ประเภท	Desktop application	Backend	Front/Back end	Front end	Front end
ความสามารถทางทฤษฎีกราฟ	ได้บางส่วน เน้นVisualization	ดีมาก	ดี	ปานกลาง	น้อย
ความสามารถทางกราฟ Visualization	ดีมาก	แย่	ดี	ดี	ปานกลาง
ความง่าย	ง่ายมาก	ง่าย	ง่าย	ยาก	ยากมาก
Document	ดีมาก	ดีมาก	ดีมาก	แย่	ดีมาก
ความเร็วในการแสดง Visualization	ดีมาก	แย่	ดี	ดี	ดี

ตารางเปรียบเทียบเครื่องมือ Network Analysis แบบต่างๆ

ในการใช้งานจริง ถ้าต้องการทำ Network analysis ที่มีความซับซ้อนก่อนส่งแสดงผลให้ user อาจจะทำการ preprocess ข้อมูลที่ Backend ก่อน ด้วย Cytoscape.js หรือ NetworkX และส่งเป็นข้อมูล JSON ไปแสดงที่ Cytoscape.js ก็ได้

Layout ที่แสดงความสัมพันธ์ของ Community composition ได้ดี มี layout แบบ Force atlas 2 ของ Gephi และ Cose ของ CytoscapeJS จากการทดลอง ผลลัพธ์ที่ได้จากการทำ Network visualization บนเว็บไซต์

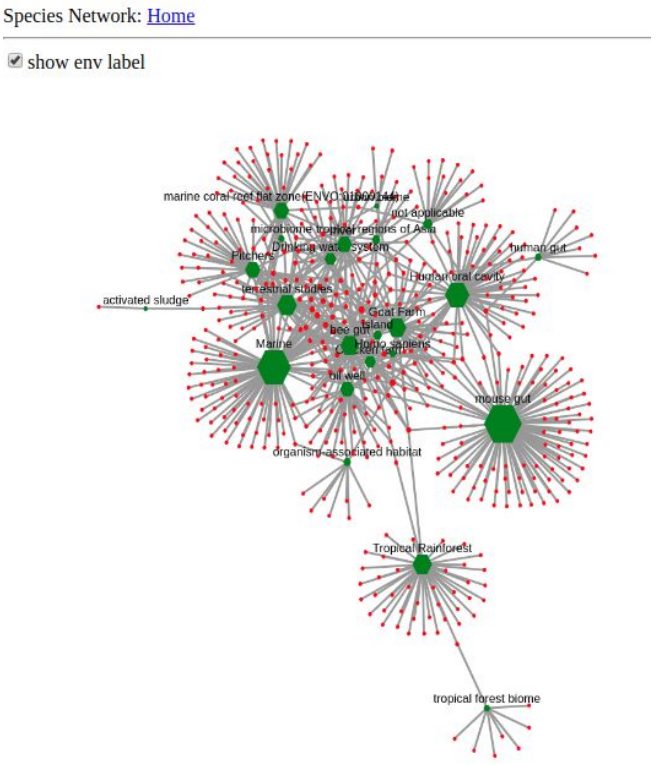
ด้วย Cytoscape.js ข้อมูลเป็นสัดเป็นส่วนไม่ชัดเจนเท่าผลลัพธ์ที่ได้จาก Gephi (ดังตัวอย่างด้านล่าง) อีกทั้งการแสดง Network visualization บน gephi ยังสามารถแสดงจำนวน node และ edge ได้มากกว่าการแสดงผลบนเว็บไซต์ (เช่น จากการทดลองที่ 17,114 nodes และ 52,772 edges Gephi ใช้เวลาประมาณ < 1 นาที ในการ render ขณะที่การ render ข้อมูลชุดเดียวกัน บน Cytoscape ใช้เวลา > 10 นาที อีกทั้งยังมีการกระตุกอย่างมากขณะแสดงผลลัพธ์บนเว็บไซต์)



ผลลัพธ์จาก Gephi layout Force Atlas 2 (ซ้าย) และ Cytoscape layout Cose (ขวา)

ไฟล์อยู่ที่ : โฟลเดอร์ docker_species/ บน CD ที่แนบมากับรายงาน

คือเว็บไซต์ที่ใช้แสดงความสัมพันธ์ของ Species และ Environment ในรูปแบบของ Network ดังรูปตัวอย่างด้านล่าง หรือ http://10.227.85.196:8000/



ผลลัพธ์จากเว็บไซต์

จุดสีแดงแสดง Species จุดสีเขียวแสดง Environment กราฟนี้จะแสดงถึงความสัมพันธ์ระหว่าง Species และ Environment ต่างๆ ซึ่งตัวอย่างของการวิเคราะห์อาจจะมีได้ดังนี้

1. Species ใดๆ อยู่ใน Environment ไหนบ้าง
2. แนวโน้มของ Environmental habitat ที่มี Community composition ใกล้เคียงกัน

Technology ที่ใช้พัฒนากับชุดข้อมูลสาธิต

1. Python ใช้สำหรับทำ Backend โดยมี Libraries หลักดังนี้
    a. Flask = framework สำหรับสร้าง Web application
    b. Pandas = สำหรับจัดการกับ CSV
2. Javascript ใช้สำหรับทำ Network ที่แสดงผลบน Browser มี Libraries หลักดังนี้
    a. Cytoscape.js = framework สำหรับทำ Graph Analysis
    b. Qtip (cytoscape plugin) = ใช้สำหรับทำการแสดงชื่อ Node เวลาเมาส์ไปทับ
3. Jinja 2 เป็น Frontend Engine ที่มาพร้อม Flask
4. Docker เป็น Virtualization สำหรับ Deployment

ชุดข้อมูลที่ใช้

เว็บไซต์ ณ ปัจจุบัน ใช้วิธีการอ่านไฟล์บน docker_species/app/static/data/bio_sra.csv ดังข้อมูลที่แสดงด้านล่าง ซึ่งข้อมูลชุดนี้มีขนาด 695,456 row ประกอบด้วย 17,114 Nodes และ 52,772 edges ซึ่งเมื่อทดลองใช้ข้อมูลทั้งหมดในการสร้างเว็บไซต์ด้วย Cytoscape.js เกิดปัญหาคือโหลดช้า และประสิทธิภาพไม่ดี ผู้เขียนจึงลดขนาดข้อมูลเหลือ 1,000 rows ประกอบด้วย 681 nodes และ 684 edges

SRA	SampleName	Bioproject	Title	BioSampleModel	env_biome

env_feature	env_material	host	collection_date	lat_lon	Country	Location
Taxon rank	superkingdom	kingdom	phylum	class	order	family	genus	species
readcounts								
SRS3032325	soil metagenome	PRJNA437389	Intensive tropical land use changes correlate with massive shifts in soil fungal communities	MIGS/MIMS/MIMARKS.soil	soil	oil palm plantation	bulk soil	Nov-2012	1.90989 S 103.26619 E	Indonesia	Jambi	Trichoderma atroviride IMI 206040	[strain]	Eukaryota	Fungi	Ascomycota	Sordariomycetes	Hypocreales	Hypocreaceae	Trichoderma	Trichoderma atroviride	56
SRS1810854	chicken gut metagenome	PRJNA353745	Caecum microbial diversity for young broiler	MIGS/MIMS/MIMARKS.host-associated	Chicken farm	Intestine	caecum content	Gallus gallus	08-Dec-2015	2.984193 N 101.728232 E	Malaysia	UPM, Serdang	cellular organisms	6								

ตัวอย่างข้อมูล bio_sra.csv

วิธีการนำตัวอย่างสคริปต์ Visualization ไป implement

โปรเจคนี้ถูกสร้างขึ้นด้วย Python Flask และนำไปวางลงบน Sever ด้วย Docker วิธีการติดตั้งแอพพริเคชั่นดังนี้

1. Copy-Paste โฟลเดอร์ docker_species/ จาก CD ที่แนบมาพร้อมกับรายงาน ไปไว้บน Server
2. ติดตั้ง Docker บนเครื่อง Server (ในกรณีที่ยังไม่ได้ติดตั้ง)
3. ใช้ Command line ไปที่โฟลเดอร์บนเครื่อง Server
4. ใช้คำสั่งด้านล่างเพื่อสร้าง Docker image ของแอพพริเคชั่นนี้

```
sudo docker build -t species_web:latest .
```

5. ใช้คำสั่งด้านล่างเพื่อรันแอพพริเคชั่นนี้ ด้วย Docker (port 8000 สามารถเปลี่ยนเป็นเลข port ที่ต้องการได้)

```
sudo docker run --name species_web -d -p 8000:5000 --rm species_web:latest
```

6. เชื่อมต่อเว็บโดย <ip เครื่อง>:port

ข้อจำกัด

จากการทดลองสร้างเว็บไซต์สำหรับการทำ Network Visualization บนเครื่องคอมพิวเตอร์ระบบปฏิบัติการ Ubuntu 18.04, Ram 8 GB, cpu i7-4790 @ 3.6GHz การแสดงผล Network Visualization บนหน้าเว็บ Browser มีข้อจำกัดเรื่องปริมาณข้อมูล 2 ด้าน คือ

1. การส่งข้อมูลจาก Backend ไปยัง Frontend ถ้าข้อมูลมีขนาดใหญ่ จะใช้เวลาในการส่งนาน
2. การ Render ข้อมูลบนเว็บ Browser มีข้อจำกัดที่ขนาดของแรมของเครื่องที่ Render

ถ้าต้องการแสดงผลลัพธ์บนเว็บจริงๆ ควรลดขนาดข้อมูลที่จะ render ในหน้าเว็บไม่ให้มากจนเกินไป (Node < 2,000) และทำ Optimization ก่อนจะมีการส่งข้อมูลจาก Backend ไป Frontend เพื่อลดเวลาในการดาวโหลดข้อมูล

แนวทางการพัฒนาโปรเจคในอนาคต

1. ในกรณีที่ต้องการวิเคราะห์ Big Graph แบบ Offline อาจจะใช้เครื่องที่เป็น Parallel computing เช่น Spark เพื่อลดเวลาในการทำงาน
2. การแสดงกราฟบนเว็บไซต์ อาจจะใช้วิธีการแสดงแค่เพียงสภาพแวดล้อม และให้ผู้ใช้เลือกสภาพแวดล้อมที่สนใจเพื่อดึงข้อมูล Species มาอีกทีหนึ่ง วิธีนี้จะลดจำนวนข้อมูลลงเยอะมาก แต่ต้องหาวิธีการเก็บข้อมูลด้านหลังให้มีประสิทธภาพ รวมถึงออกแบบหน้าตาเว็บไซต์ให้ดีด้วย

สรุป

งานนี้ใช้วิเคราะห์จำนวน Gene, species, และ Protein functions จากข้อมูล metagenome ของ Microbiome project อีกทั้งโปรเจคนี้ ได้ทำระบบการสกัดและสร้างฐานข้อมูล Kegg Organism Gene ไว้บน Server HPC ของ Biotec เพื่อใช้ประกอบกับไปป์ไลน์ในส่วนแรกในการวิเคราะห์ Protein Function และท้ายที่สุดผู้เขียนได้ทำการวิเคราะห์ข้อมูล Microbiome ชุดตัวอย่างด้วย Network Visualization เพื่อแสดงความสัมพันธ์เบื้องต้นของ Environmental habitat กับ Microbial community composition อีกด้วย