TF Kandidaten für ein Gen finden und verifizieren

Daten

- Multiple Sequence Alignment, Maus als Referenz, Spalax unter den Queries
    - Speicherort Samrock, /moreAddSpace/abrzoska/mm10/multi.anno.maf
    - ❗ Bei allen Annotationen verifizieren, dass sie auf der selben Assembly (hg38/mm10) beruhen.
- Annotation (cCRE/Encode, CNEs, etc. in .bed-Format)

1. Annotation filtern auf upstream-Region des zu untersuchenden Gens

    ```
    reg_ex = rex.RegionExtractor()
    reg_ex.find_bed_candidates(bfile, start, end, scaffold, output_dir)
    ```

    bfile: Annotation in .bed format

    start/end/scaffold: Genomausschnitt, e.g. *start=86700000, end=86750000, scaffold='chr17'*

    Output: .bed mit Regionen, die Annotationen in der angegebenen Region repräsentieren

2. TFmotifView Kandidaten generieren

    http://bardet.u-strasbg.fr/tfmotifview/

    Assembly selektieren, Genomregionen aus 1. hochladen/einfügen, starten. Ergebnisse kommen per Mail/Link

3. Erstellen der ENCODE TF Liste

    https://screen.wenglab.org/

    Gen suchen, cCRE eins nach dem anderen auswählen, unter TF and His-mod Intersection die TF rauskopieren und in eine Liste einfügen (Jedes auf neuer Zeile, Duplikate sind kein Problem)

factor	# of experiments that support TF binding	# experiments in total
POLR2A	14	16
CTCF	5	31
HCFC1	3	3
MAZ	2	2
TBP	2	2

4. Vergleich

mit der Liste aus 2) und der Liste aus 3) kann nun mithilfe des tftranslate.py Skript sortiert werden, welche Übereinstimmungen es gibt, unabhängig von TF-Aliasen
Schritte:

- mit encode liste (2)
    - Eventuelle Duplikate, die beim rauskopieren entstanden sind: remove_duplicates(infile, outfile)
    - Alias-Dictionary erstellen: get_dict_from_gene_list(outfile, dict)
- mit tfmotifview liste
    - bed Datei zuschneiden (sodas nur TF drin steht: get_tfmotifview_tfs_from_bed(tfmv_in, tfmv_tf)
    - Duplikate entfernen: remove_duplicates(tfmv_tf, tfmv_out)
- Mit Ergebnissen überprüfen welche Übereinstimmungen durch Abgleich mit Alias-Namen zum Vorschein treten: check_which_tfs_are_in_common(dict, tfmv_out, comparison_result)

Beispiel-Output:

Present in both sets (True positive)

TBP (GTF2D,GTF2D1,HDL4,SCA17,TFIID)
HCFC1 (CFF,HCF,HCF-1,HCF1,HFC1,MRX3,PPP1R89,VCAF)
..

Only in dictionary, based on experiments (False Negative)

CHD1 (CHD-1,PILBOS)
TCF12 (CRS3,HEB,HTF4,HsT17266,TCF-12,bHLHb20,p64)
…

Only in predictions, based on algorithms (possible False Positive)

RORC
DPRX
…