IR Assignment 3

Group – Fantastic Four
Members:

1. Rahul Kumar S20160010069
2. Venkat Lokesh S20160010078
3. Nikhil Singh S20160010103
4. Y. Hemanth S20160010108

Movie Name 1: Operation Mad Ball
Movie Name 2: Dragon Ball Super
Movie Name 3: Manmadhudu A Ball Boy

1. 3-gram Non-positional Inverted Index:

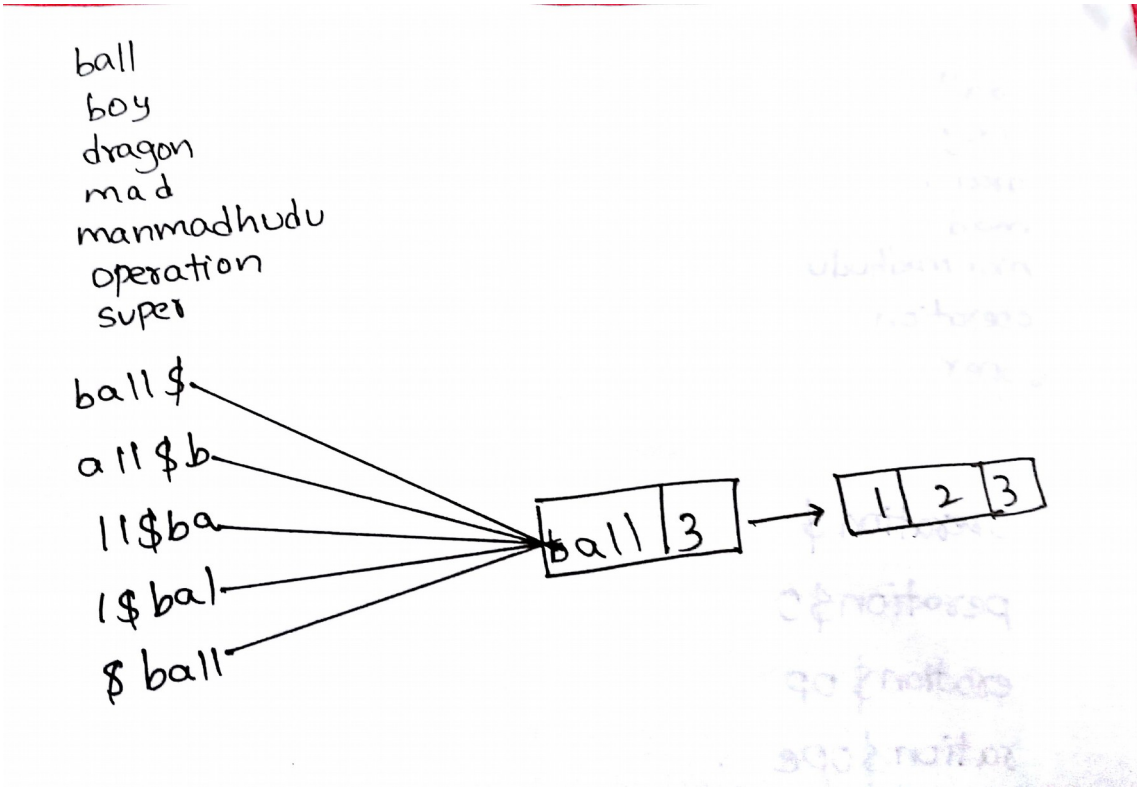
Words after case folding and stop words removal:-

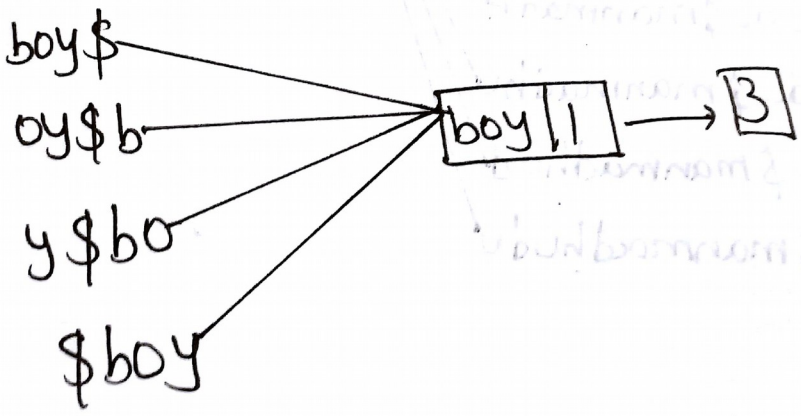
1. ball
2. boy
3. dragon
4. mad
5. manmadhudu
6. operation
7. super

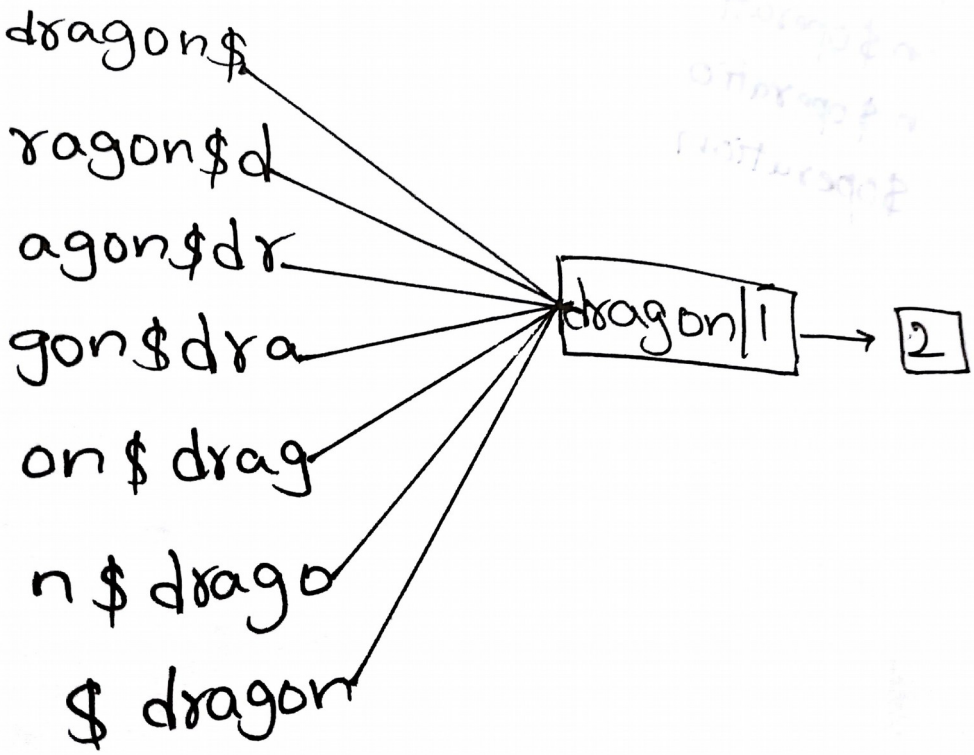
tri-gram	tri-gram freq	docId	WORD
$ba	1	1,2,3	ball
$bo	1	3	boy
$dr	1	2	dragon
$ma	2	1,3	manmadhudu, mad
$op	1	1	operation
$su	1	2	Super
ad$	1	1	mad
adh	1	3	manmadhudu
ago	1	2	dragon
all	1	1,2,3	ball
anm	1	3	manmadhudu
ati	1	1	Operation
bal	1	1,2,3	ball
boy	1	3	boy
dhu	1	3	manmadhudu
dra	1	2	dragon
du$	1	3	manmadhudu
er$	1	2	super

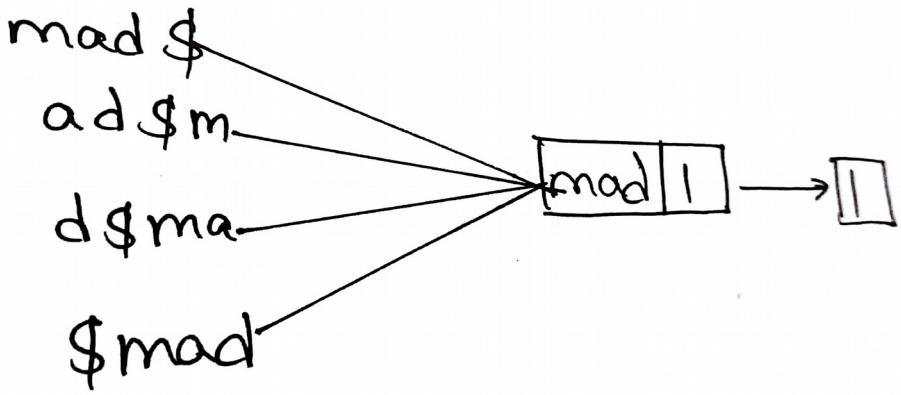
tri-gram	tri-gram freq	docId	WORD
era	1	1	Operation
gon	1	2	dragon
hud	1	3	manmadhudu
ion	1	1	operation
ll$	1	1,2,3	ball
mad	2	1,3	manmadhudu, mad
man	1	3	manmadhudu
nma	1	3	manmadhudu
on$	2	1,2	operation, dragon
ope	1	1	operation
oy$	1	3	boy
per	2	1,2	operation, super
rag	1	2	dragon
rat	1	1	operation
sup	1	2	Super
tio	1	1	operation
udu	1	3	manmadhudu
upe	1	2	Super

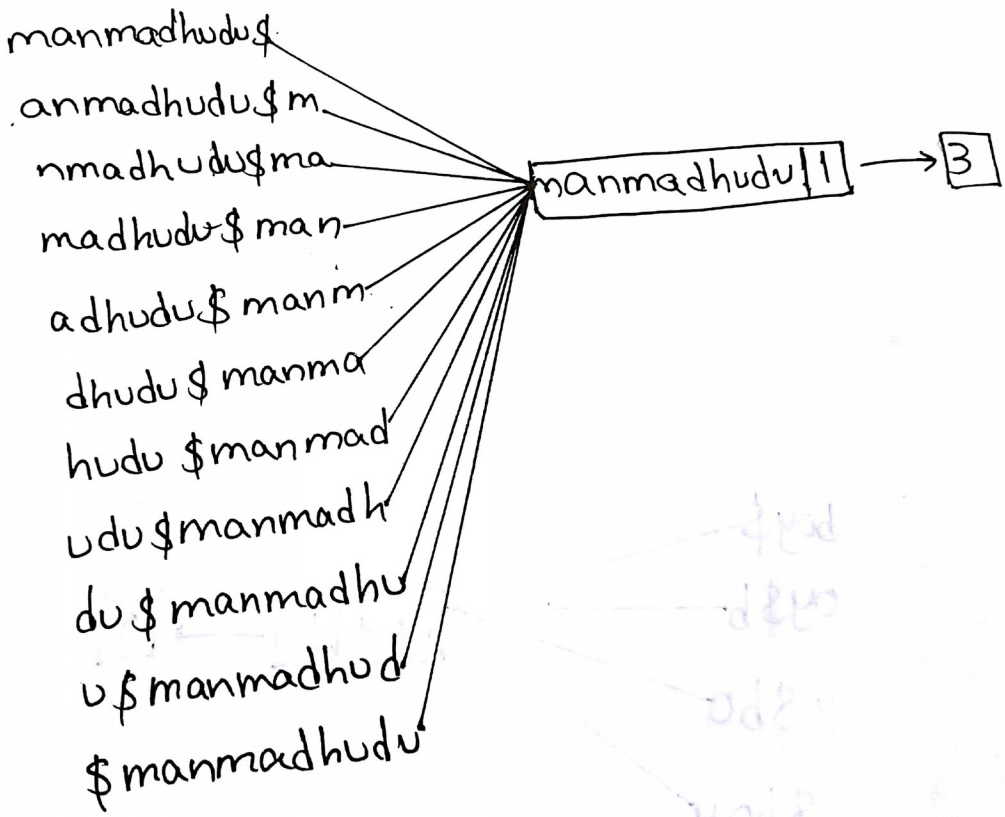
2. Permuterm Non-positional Inverted index:

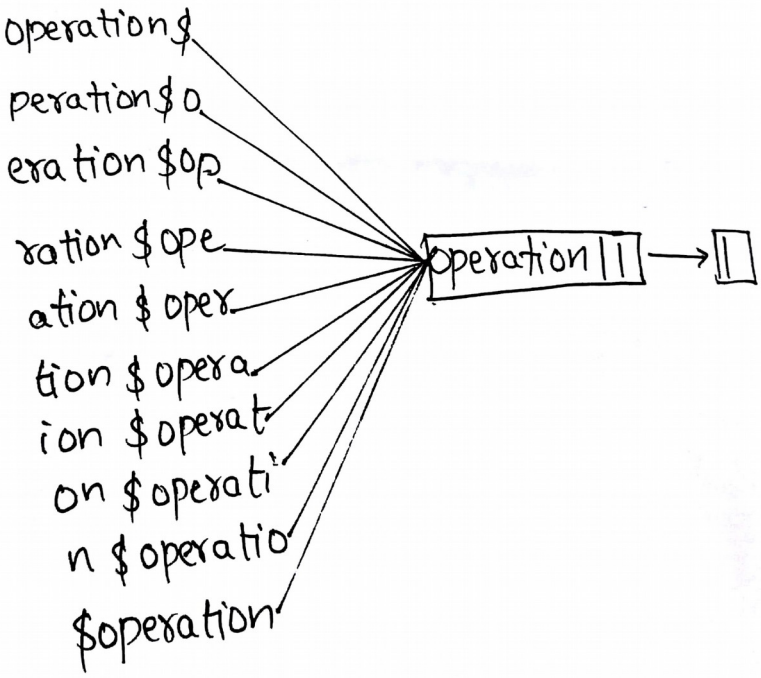


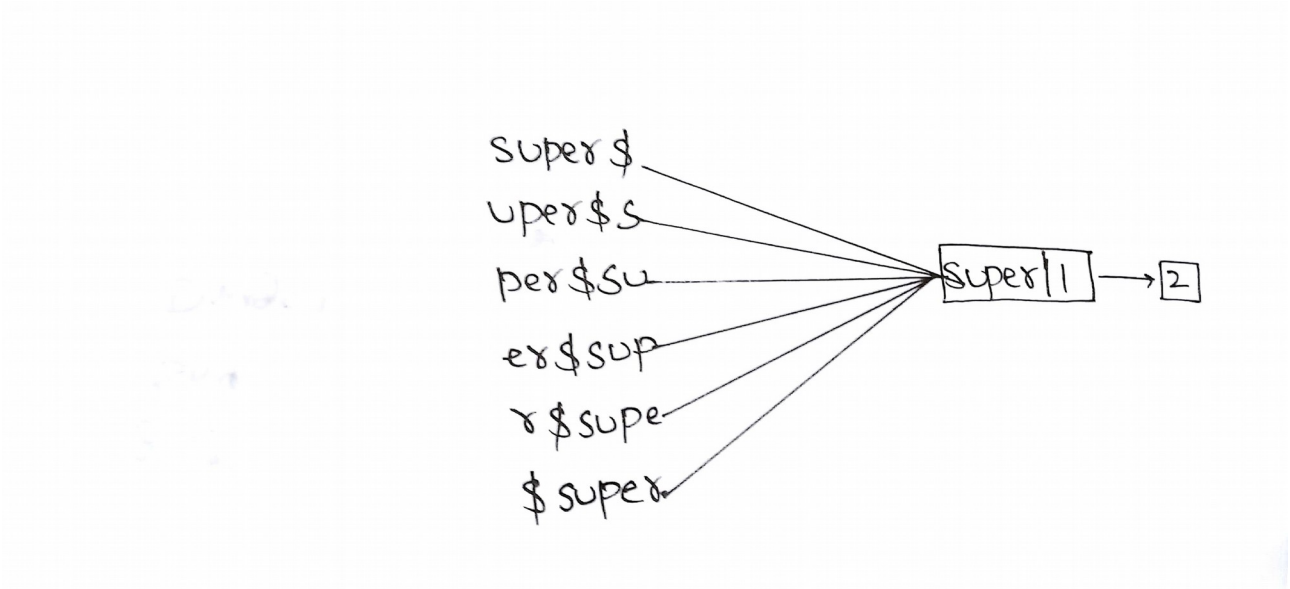



dragon$
ragon$d
agon$dr
gon$dra
on$drag
n$drago
$dragon
dragon 1
2


mad$
ad$m
d$ma
$mad
mad 1
1


manmadhudu$
anmadhudu$m
nmadhudu$ma
madhudu$man
adhudu$manm
dhudu$manma
hudu$manmad
udu$manmadh
du$manmadhu
u$manmadhud
$manmadhudu
manmadhudu|1
3


operation$
peration$o
eration$op
ration$ope
ation$oper
tion$opera
ion$operat
on$operati
n$operatio
$operation
operation|1
1



3. Explanation:

Query1:dra*n

permuterm index:

How does this index help us with wildcard queries? Consider the wildcard query **dra*n**. The key is to rotate such a wildcard query so that the * symbol appears at the end of the string - thus the rotated wildcard query becomes **n$dra***. Next, we look up this string in the permuterm index, where seeking **n$dra*** (via a search tree) leads to rotations of (among others) the term dragon.which gives **document 2** as output.

Now that the permuterm index enables us to identify the original vocabulary terms matching a wildcard query, we look up these terms in the standard inverted index to retrieve matching documents. We

can thus handle any wildcard query

3-gram index:
How does such an index help us with wildcard queries? Consider the wildcard query **dra*n**. We are seeking documents containing any
term that begins with **dra** and ends with **n**. Accordingly, we run the
Boolean query **$dr** AND **dra**. This is looked up in the 3-gram index
and yields a list of matching term dragon. which gives **document 2** as output.
Each of these matching terms is then looked up in the standard inverted
index to yield documents matching the query.

Query2:mad*
permuterm index:
mad*$** is converted into **$mad* now it looks the permuterm index

for terms matching with $mad* and leads to the term mad.which gives document 1.
3-gram index:
mad* is divided into 3 grams **$ma** and **mad**.this gives two terms **manmadhudu** and **mad**.In which manmadhudu term doesn't match the
given query.which is a mismatch.
it retrieves document 1(true positive),document 3(false positive)

The permuterm index is simple but, it can lead to a considerable blowup from the number of rotations per term; for a dictionary of English terms, this can represent an almost ten-fold space increase.

In k-gram index some times there will be a mismatch between query
and result.for example see how it performed in Query2.To cope with this,
we introduce a post-filtering step, in which the terms enumerated by the
Boolean query on the 3-gram index are checked individually against the original query
mad*. This is a simple string-matching operation and weeds out terms such as **manmadhudu**
that do not match the original query. Terms that survive are then searched in
the standard inverted index as usual.

k-gram index is better as it takes lesser space than permuterm and with postfiltering we can overcome mismatching problem in k-gram index.