Blending Levels from Different Games using LSTMs

Anurag Sarkar and **Seth Cooper**

College of Computer and Information Science

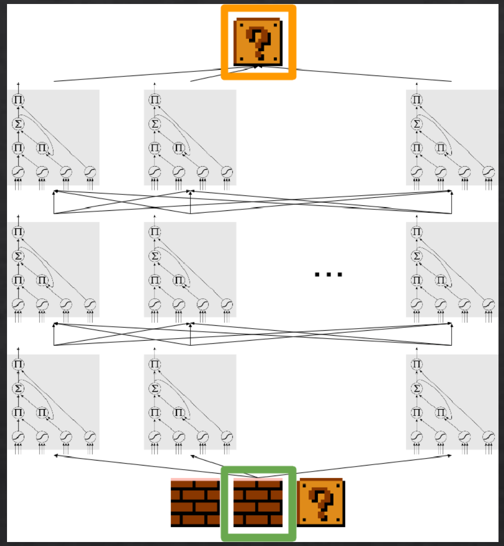
Northeastern University

Motivation

- Recent work on training models on existing levels to generate new levels

Motivation

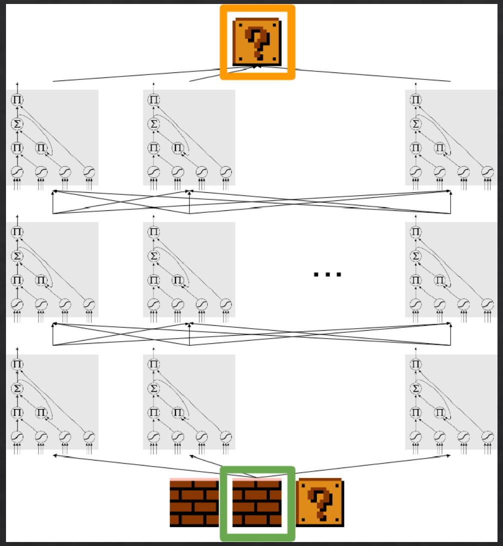
- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs



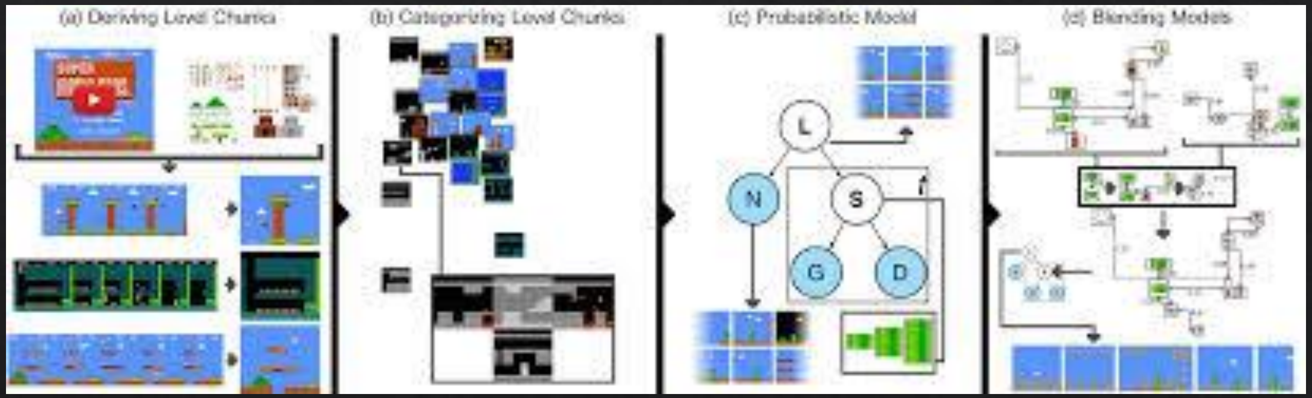
Summerville and Mateas, 2016

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Conceptual blending



Summerville and Mateas, 2016



Guzdial and Riedl, 2016

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Conceptual blending
- Gow and Corneli proposed generating new games by blending entire games

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Conceptual blending

- Gow and Corneli proposed generating new games by blending entire games

```
SpriteSet
  forest > SpawnPoint stype=log
    dense > prob=0.4 cooldown=10
      sparse > prob=0.1 cooldown=5
  structure > Immovable
      home > Door
      water
      wall
  log > Missile orientation=LEFT speed=0.1
  safety > Resource limit=2
  frog > MovingAvatar
  truck > Missile img=truck
      rightTruck > orientation=RIGHT
          fastRtruck > speed=0.2
          slowRtruck > speed=0.1
      leftTruck > orientation=LEFT
          fastLtruck > speed=0.2
          slowLtruck > speed=0.1

InteractionSet
  home frog  > killSprite scoreChange=1
  frog log   > changeResource resource=safety value=2
  frog log   > pullWithIt
  frog wall  > stepBack
  frog water > killIfHasLess resource=safety limit=0
  frog water > changeResource resource=safety value=-1
  frog truck > killSprite scoreChange=-2
  log   EOS  > killSprite
  truck EOS  > wrapAround

TerminationSet
  SpriteCounter stype=home win=True
  SpriteCounter stype=frog win=False
```

VGDL Frogger

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Conceptual blending
- Gow and Corneli proposed generating new games by blending entire games

```
SpriteSet
  forest > SpawnPoint stype=log
    dense > prob=0.4 cooldown=10
      sparse > prob=0.1 cooldown=5
  structure > Immovable
      home > Door
      water
      wall
  log > Missile orientation=LEFT speed=0.1
  safety > Resource limit=2
  frog > MovingAvatar
  truck > Missile img=truck
      rightTruck > orientation=RIGHT
          fastRtruck > speed=0.2
          slowRtruck > speed=0.1
      leftTruck > orientation=LEFT
          fastLtruck > speed=0.2
          slowLtruck > speed=0.1

InteractionSet
  home frog  > killSprite scoreChange=1
  frog log   > changeResource resource=safety value=2
  frog log   > pullWithIt
  frog wall  > stepBack
  frog water > killIfHasLess resource=safety limit=0
  frog water > changeResource resource=safety value=-1
  frog truck > killSprite scoreChange=-2
  log   EOS  > killSprite
  truck EOS  > wrapAround

TerminationSet
  SpriteCounter stype=home win=True
  SpriteCounter stype=frog win=False
```

VGDL Frogger

```
SpriteSet
  door > Door
  key > Immovable
  wall > Immovable
  sword > Flicker limit=5 singleton=True
  movable >
      link > ShootAvatar stype=sword
          nokey
          withkey
      monster > RandomNPC
          monsterQuick > cooldown=2
          monsterNormal > cooldown=4
          monsterSlow > cooldown=8

InteractionSet
  movable wall  > stepBack
  nokey door    > stepBack
  door withkey  > killSprite scoreChange=1
  monster sword > killSprite scoreChange=2
  link monster  > killSprite scoreChange=-1
  key link      > killSprite scoreChange=1
  nokey key     > transformTo stype=withkey

TerminationSet
  SpriteCounter stype=door win=True
  SpriteCounter stype=link win=False
```

VGDL Zelda

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Conceptual blending
- Gow and Corneli proposed generating new games by blending entire games

```
SpriteSet
  forest > SpawnPoint stype=log
    dense > prob=0.4 cooldown=10
      sparse > prob=0.1 cooldown=5
  structure > Immovable
      home > Door
      water
      wall
  log > Missile orientation=LEFT speed=0.1
  safety > Resource limit=2
  frog > MovingAvatar
  truck > Missile img=truck
      rightTruck > orientation=RIGHT
          fastRtruck > speed=0.2
          slowRtruck > speed=0.1
      leftTruck > orientation=LEFT
          fastLtruck > speed=0.2
          slowLtruck > speed=0.1

InteractionSet
  home frog  > killSprite scoreChange=1
  frog log   > changeResource resource=safety value=2
  frog log   > pullWithIt
  frog wall  > stepBack
  frog water > killIfHasLess resource=safety limit=0
  frog water > changeResource resource=safety value=-1
  frog truck > killSprite scoreChange=-2
  log   EOS  > killSprite
  truck EOS  > wrapAround

TerminationSet
  SpriteCounter stype=home win=True
  SpriteCounter stype=frog win=False
```

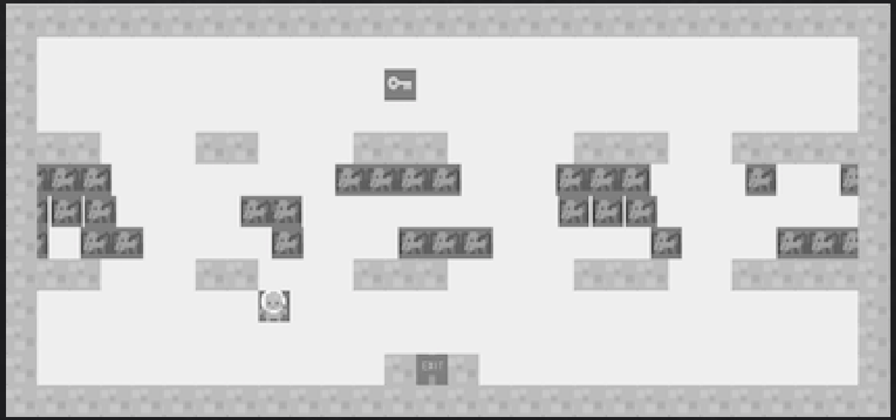
VGDL Frogger

```
SpriteSet
  door > Door
  key > Immovable
  wall > Immovable
  sword > Flicker limit=5 singleton=True
  movable >
      link > ShootAvatar stype=sword
          nokey
          withkey
      monster > RandomNPC
          monsterQuick > cooldown=2
          monsterNormal > cooldown=4
          monsterSlow > cooldown=8

InteractionSet
  movable wall  > stepBack
  nokey door    > stepBack
  door withkey  > killSprite scoreChange=1
  monster sword > killSprite scoreChange=2
  link monster  > killSprite scoreChange=-1
  key link      > killSprite scoreChange=1
  nokey key     > transformTo stype=withkey

TerminationSet
  SpriteCounter stype=door win=True
  SpriteCounter stype=link win=False
```

VGDL Zelda



Frolda

Motivation

- Recent work on training models on existing levels to generate new levels
  - Sequence Prediction using LSTMs
  - Concept
- Gow and C
  generating
  entire games

```
SpriteSet
  forest > SpawnPoint stype=log
    dense > prob=0.4 cooldown=10
      sparse > prob=0.1 cooldown=5
  structure > Immovable
      home > Door
      water
      wall
  log > Missile orientation=LEFT speed=0.1
  safety > Resource limit=2
  frog > MovingAvatar
  truck > Missile img=truck
      rightTruck > orientation=RIGHT
          fastRtruck > speed=0.2
          slowRtruck > speed=0.1
      leftTruck > orientation=LEFT
          fastLtruck > speed=0.2
```

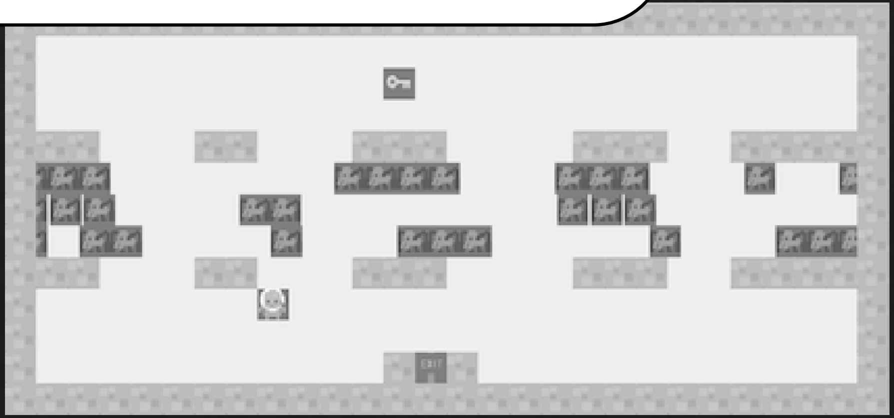
```
SpriteSet
  door > Door
  key > Immovable
  wall > Immovable
  sword > Flicker limit=5 singleton=True
  movable >
      link > ShootAvatar stype=sword
          nokey
          withkey
      monster > RandomNPC
          monsterQuick > cooldown=2
          monsterNormal > cooldown=4
          monsterSlow > cooldown=8

ionSet
 all   > stepBack
 r     > stepBack
 key   > killSprite scoreChange=1
 word  > killSprite scoreChange=2
 ter   > killSprite scoreChange=-1
       > killSprite scoreChange=1
       > transformTo stype=withkey

Set
 nter stype=door win=True
 nter stype=link win=False
```

GDL Zelda

IDEA: PCGML techniques + Game Blending



Frolda

Overview

- Train LSTMs on existing levels of *Super Mario Bros.* and *Kid Icarus*

Overview

- Train LSTMs on existing levels of *Super Mario Bros.* and *Kid Icarus*
- Sample from the trained models to generate new levels that contain properties of levels from both games

Overview

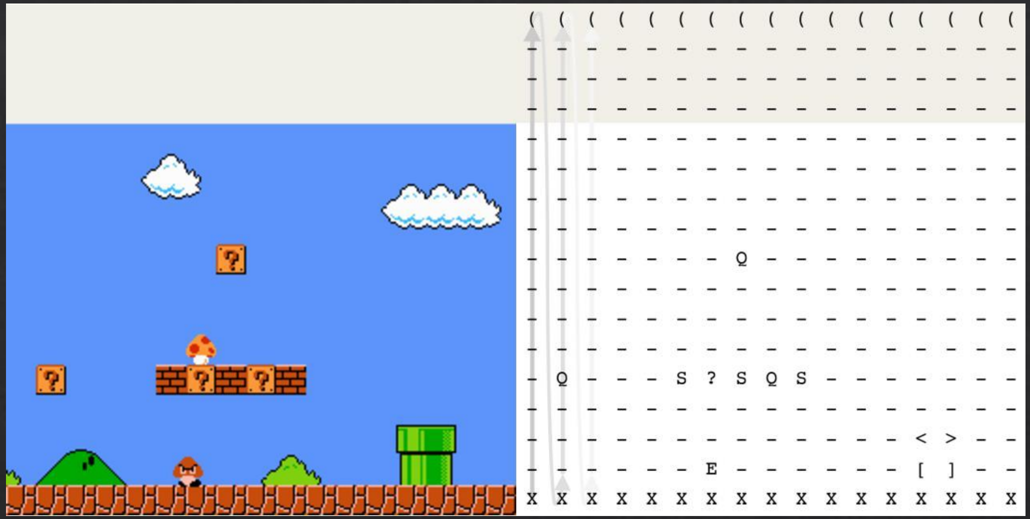
- Train LSTMs on existing levels of *Super Mario Bros.* and *Kid Icarus*
- Sample from the trained models to generate new levels that contain properties of levels from both games
- Used weights to control approximate amount of each game

Dataset

- Video Game Level Corpus (VGLC)
  - *Super Mario Bros. (15)*
  - *Kid Icarus (6)*

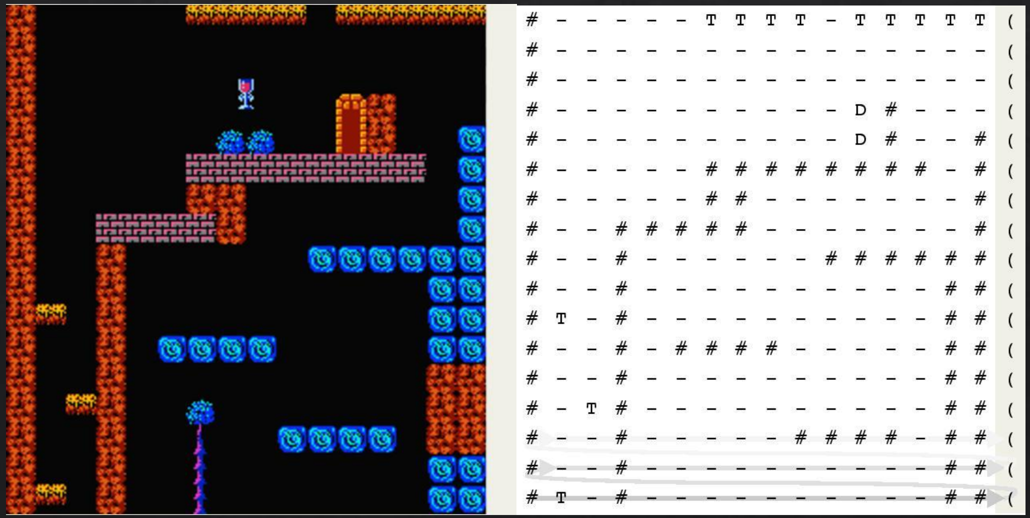
Dataset

- Video Game Level Corpus (VGLC)
  - *Super Mario Bros. (15)*
  - *Kid Icarus (6)*



SMB Level 1-1

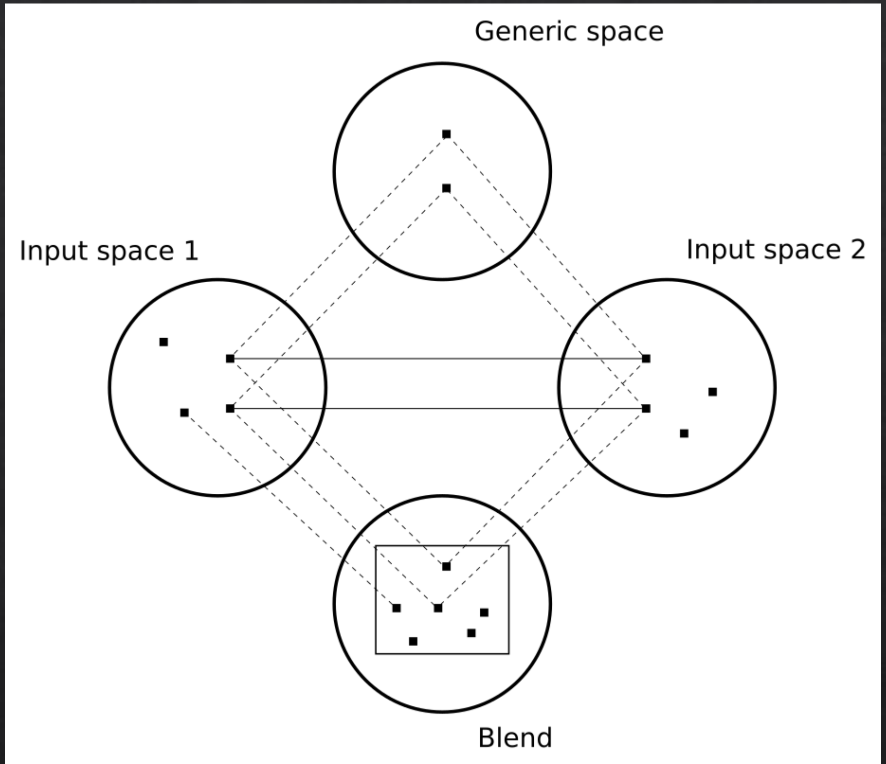
- Levels are represented as text files with each character mapping to a specific tile



KI Level 1

Blending

- Conceptual blending
  - Two input spaces
  - A generic space
  - A blend space



Blending

- Conceptual blending
  - Two input spaces
  - A generic space
  - A blend space

- Input spaces were VGLC SMB and KI corpora

```
{
    "tiles" : {
        "X" : ["solid","ground"],
        "S" : ["solid","breakable"],
        "-" : ["passable","empty"],
        "?" : ["solid","question block", "full question block"],
        "Q" : ["solid","question block", "empty question block"],
        "E" : ["enemy","damaging","hazard","moving"],
        "<" : ["solid","top-left pipe","pipe"],
        ">" : ["solid","top-right pipe","pipe"],
        "[" : ["solid","left pipe","pipe"],
        "]" : ["solid","right pipe","pipe"],
        "o" : ["coin","collectable","passable"]
    }
}
```

SMB Mapping

```
{
    "tiles" : {
        "#" : ["solid","ground"],
        "-" : ["passable","empty"],
        "D" : ["solid","openable","door"],
        "H" : ["solid","damaging","hazard"],
        "M" : ["solidtop","passable","moving","platform"],
        "T" : ["solidtop","passable","platform"]
    }
}
```

KI Mapping

Blending

- Conceptual blending
  - Two input spaces
  - A generic space
  - A blend space
- Input spaces were VGLC SMB and KI corpora
- For generic space, mapped semantically common elements to a uniform representation and preserved unique elements

```
generic_mapping = {
    # already generic
    "-": "-",

    # mario to generic
    "X": "X",
    "E": "E",
    "Q": "Q",
    "?": "?",
    "<": "<",
    ">": ">",
    "[": "[",
    "]": "]",
    "o": "o",
    "S": "S",

    # icarus to generic
    "#": "X",
    "H": "E",
    "T": "T",
    "M": "M",
    "D": "D"
    }
```

Generic Mapping

Blending

- Conceptual blending
  - Two input spaces
  - A generic space
  - A blend space
- Input spaces were VGLC SMB and KI corpora
- For generic space, mapped semantically common elements to a uniform representation and preserved unique elements
- Common elements were solid ground, enemy/hazard and the background character

```
generic_mapping = {
    # already generic
    "-": "-",

    # mario to generic
    "X": "X",
    "E": "E",
    "Q": "Q",
    "?": "?",
    "<": "<",
    ">": ">",
    "[": "[",
    "]": "]",
    "o": "o",
    "S": "S",

    # icarus to generic
    "#": "X",
    "H": "E",
    "T": "T",
    "M": "M",
    "D": "D"
    }
```

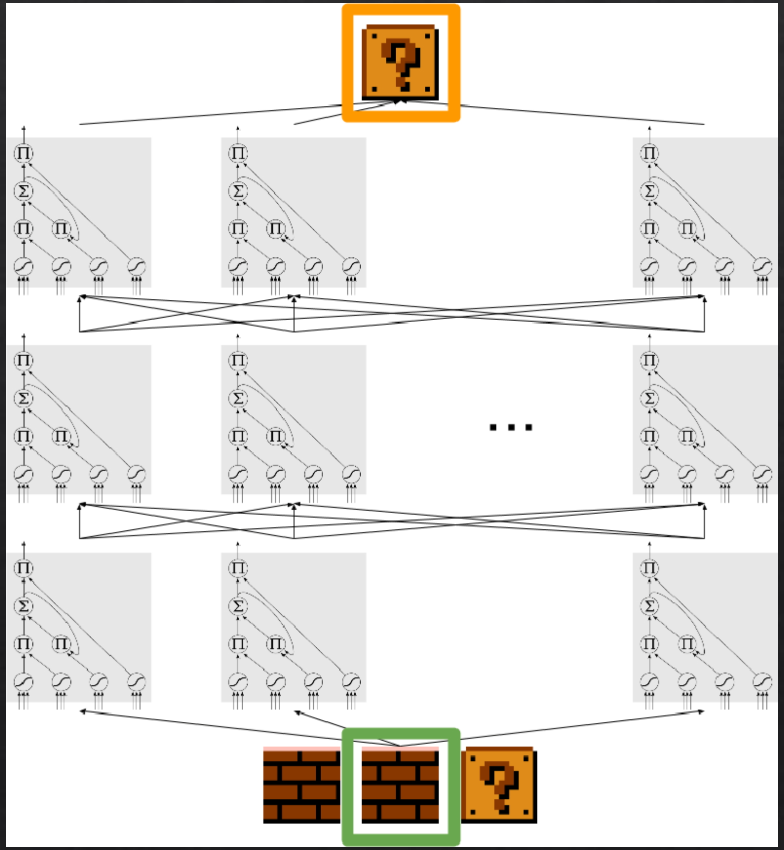
Generic Mapping

Training on Level Data

- Used Long Short Term Memory networks (LSTMs) for training

Training on Level Data

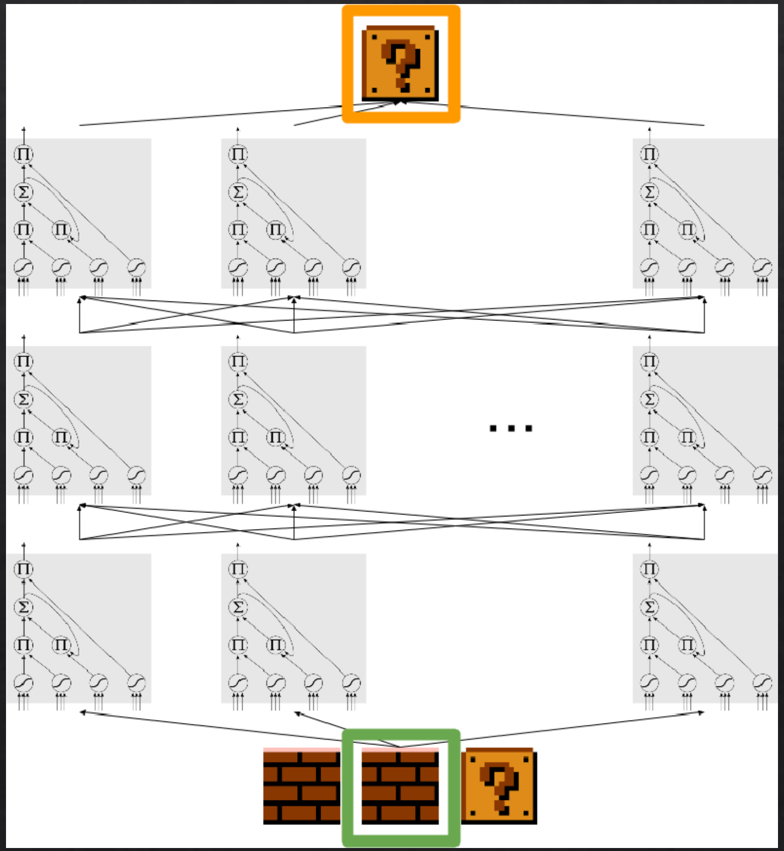
- Used Long Short Term Memory networks (LSTMs) for training
  - Predicts next item in a sequence given the sequence thus far using learned probability distribution



Summerville and Mateas, 2016

Training on Level Data

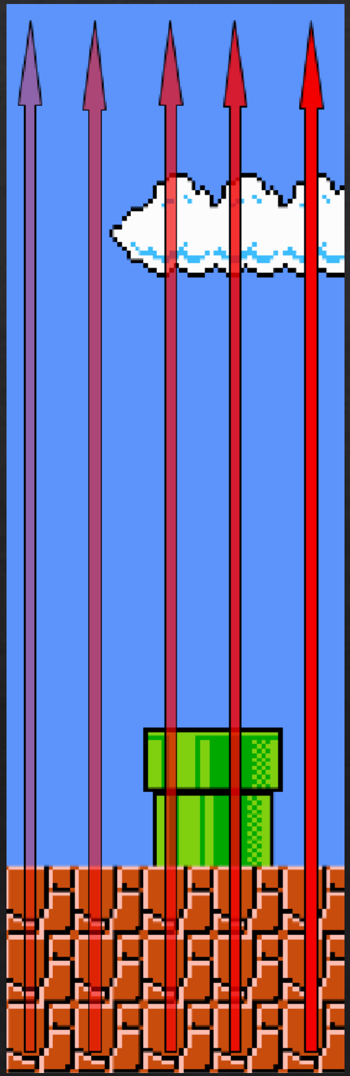
- Used Long Short Term Memory networks (LSTMs) for training
  - Predicts next item in a sequence given the sequence thus far using learned probability distribution
  - Past success in generating SMB levels



Summerville and Mateas, 2016

Training on Level Data

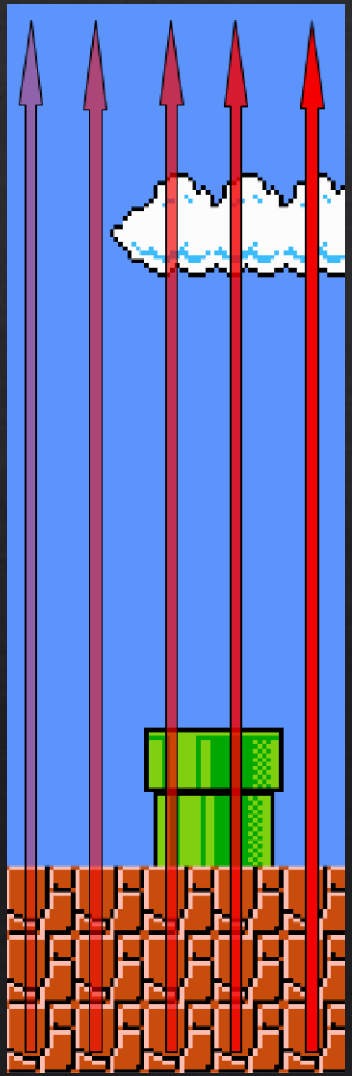
- Each level is a collection of sequences and each tile is a point in a sequence



Summerville and Mateas, 2016

Training on Level Data

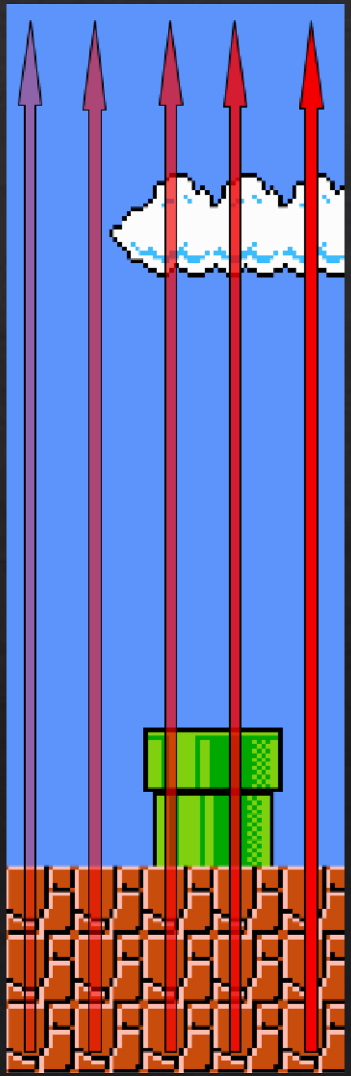
- Each level is a collection of sequences and each tile is a point in a sequence
- SMB → feed in sequences of columns from left to right
  KI → feed in sequences of rows from bottom to top



Summerville and Mateas, 2016

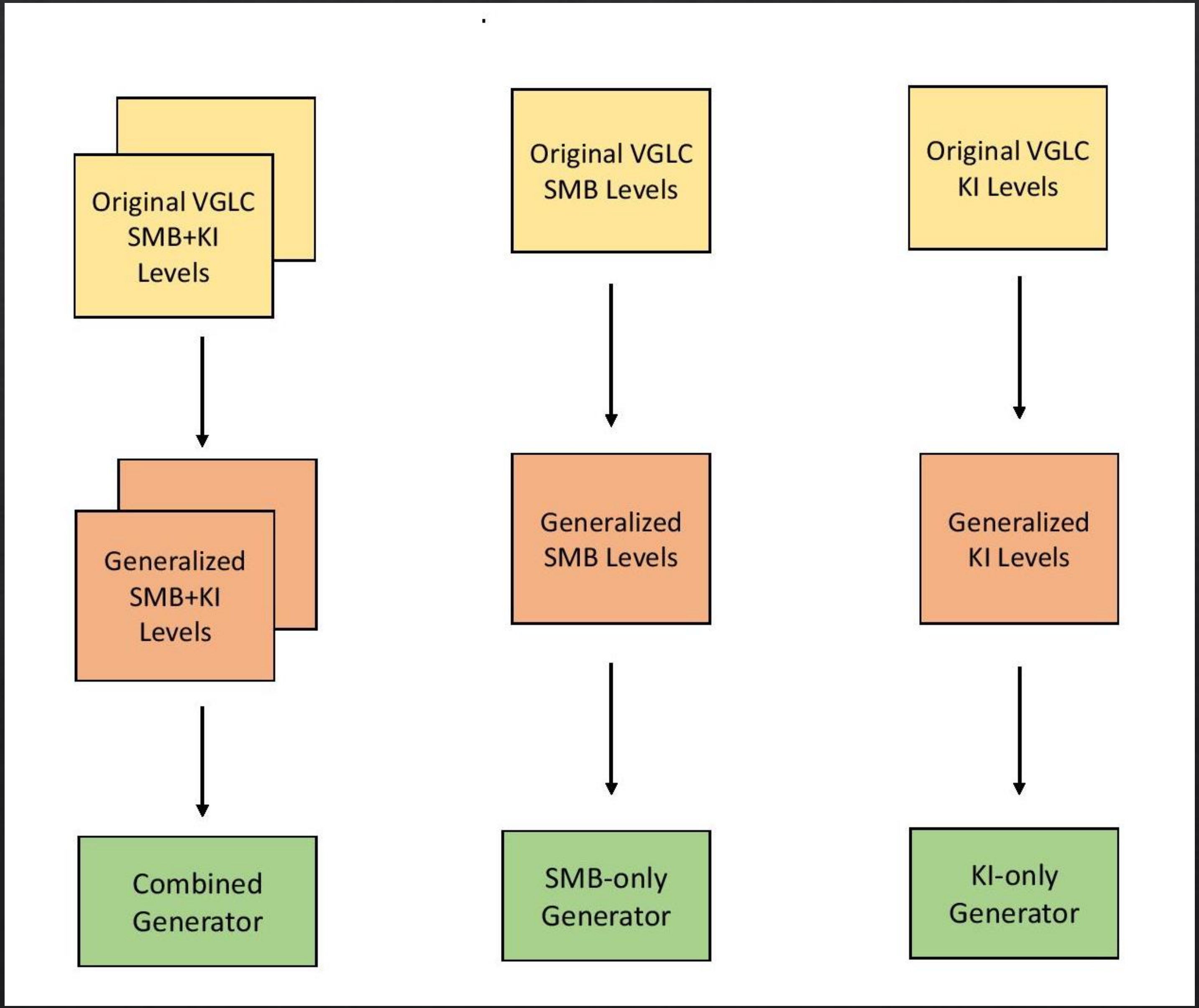
Training on Level Data

- Each level is a collection of sequences and each tile is a point in a sequence
- SMB → feed in sequences of columns from left to right
  KI → feed in sequences of rows from bottom to top
- LSTM was trained on sequences of 16 characters



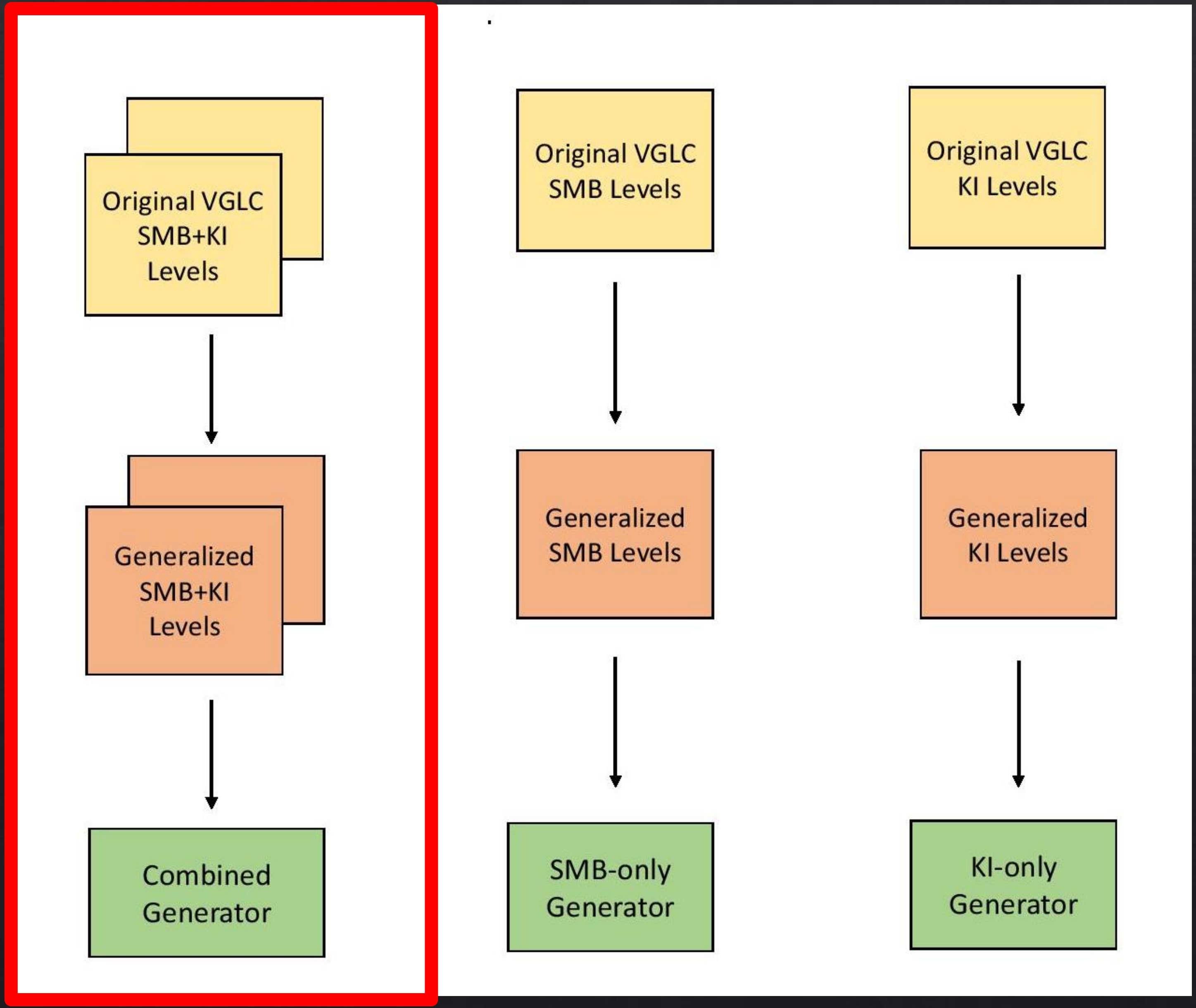
Summerville and Mateas, 2016

Models

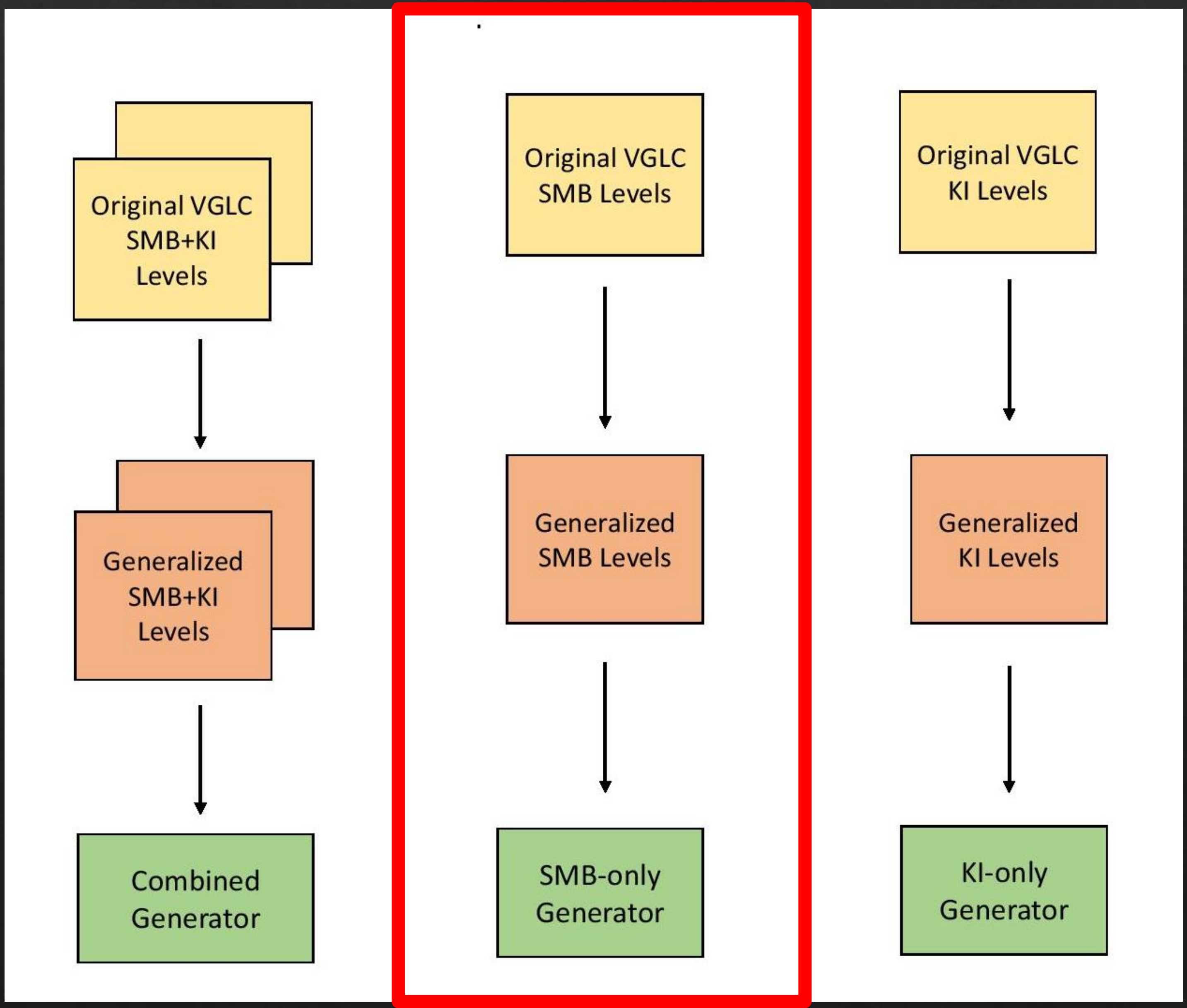


Original VGLC SMB+KI Levels
Original VGLC SMB Levels
Original VGLC KI Levels
Generalized SMB+KI Levels
Generalized SMB Levels
Generalized KI Levels
Combined Generator
SMB-only Generator
KI-only Generator

Models

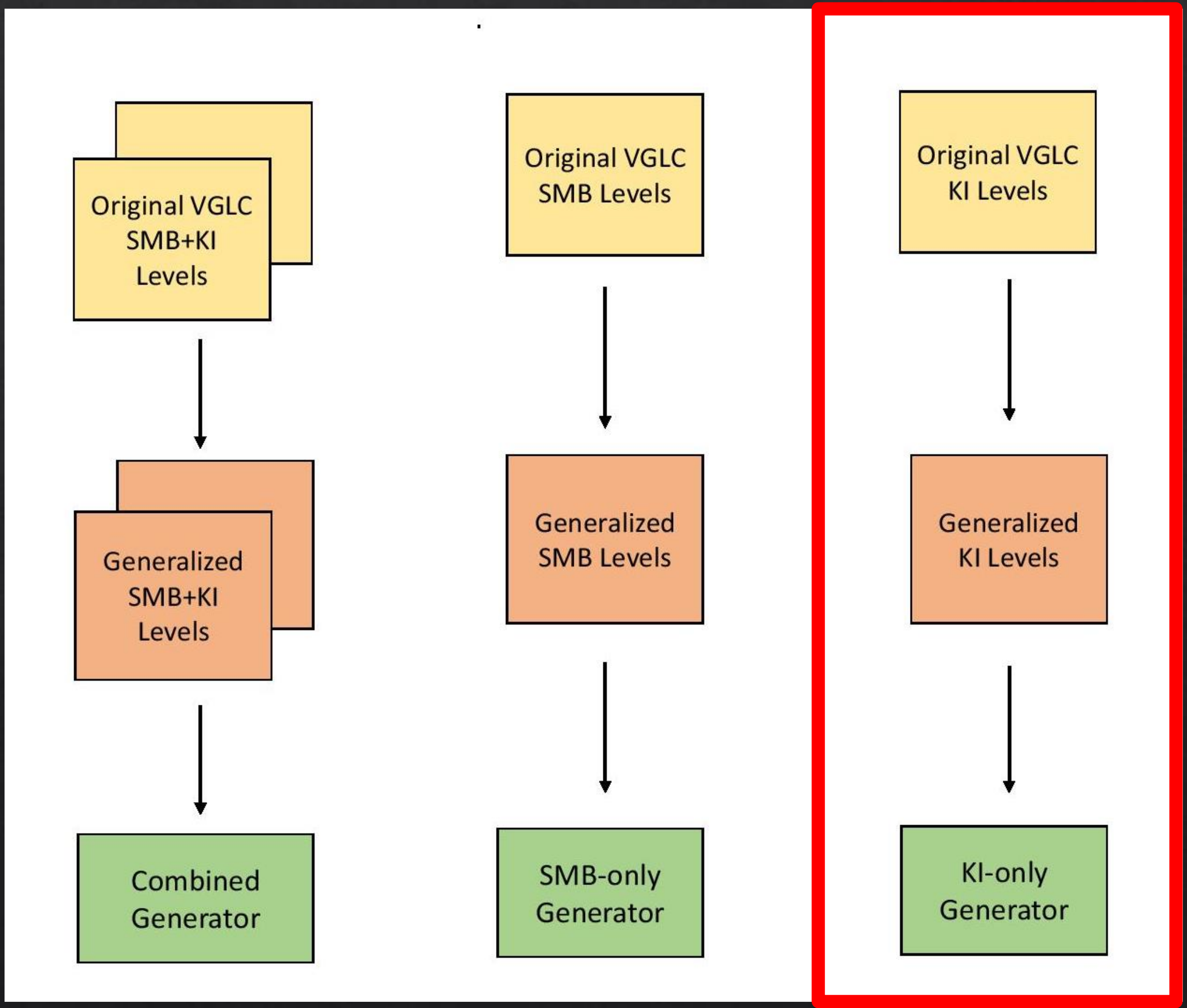

Original VGLC SMB+KI Levels
Generalized SMB+KI Levels
Combined Generator
Original VGLC SMB Levels
Generalized SMB Levels
SMB-only Generator
Original VGLC KI Levels
Generalized KI Levels
KI-only Generator

Models



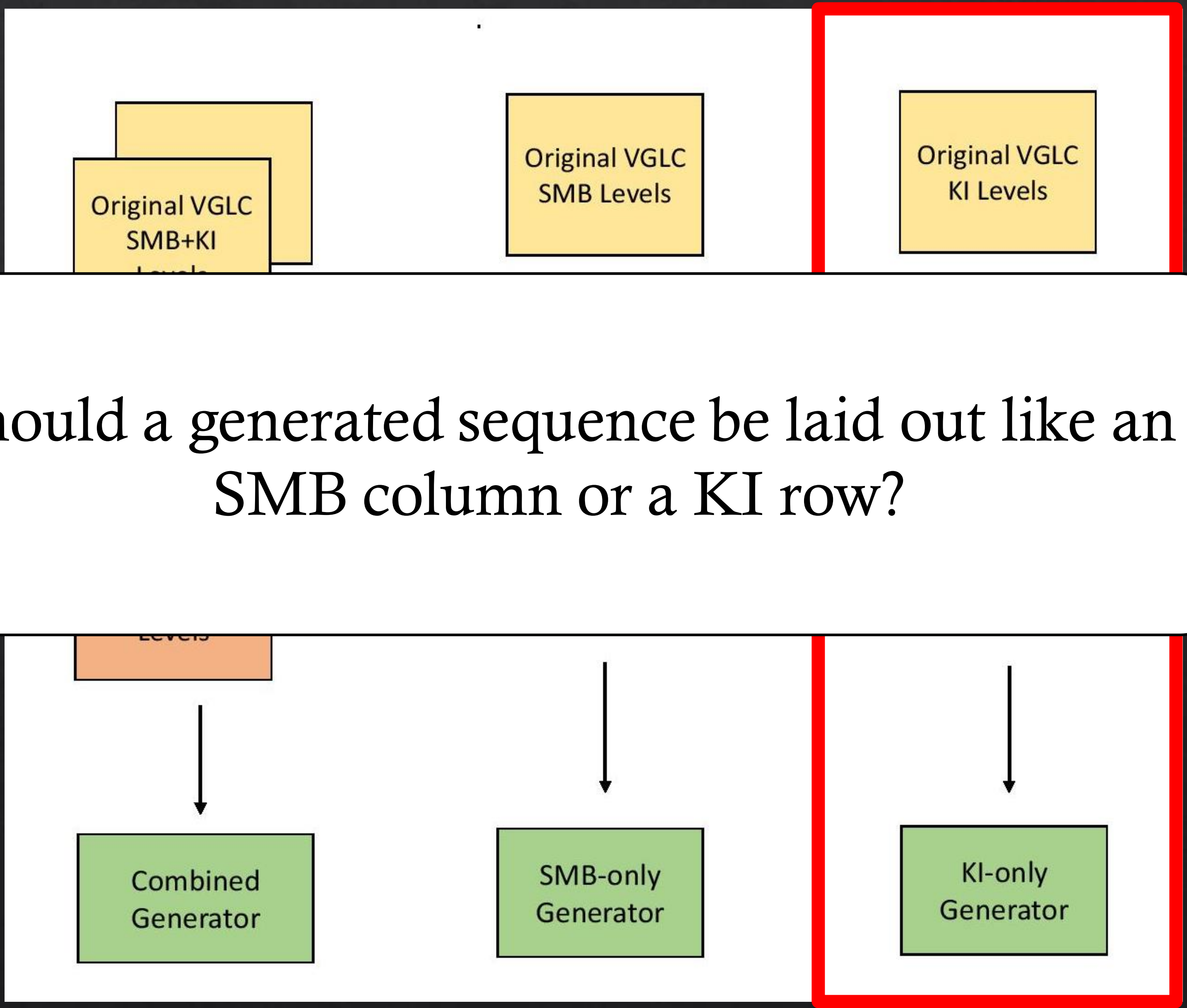
Original VGLC SMB+KI Levels
Generalized SMB+KI Levels
Combined Generator
Original VGLC SMB Levels
Generalized SMB Levels
SMB-only Generator
Original VGLC KI Levels
Generalized KI Levels
KI-only Generator

Models



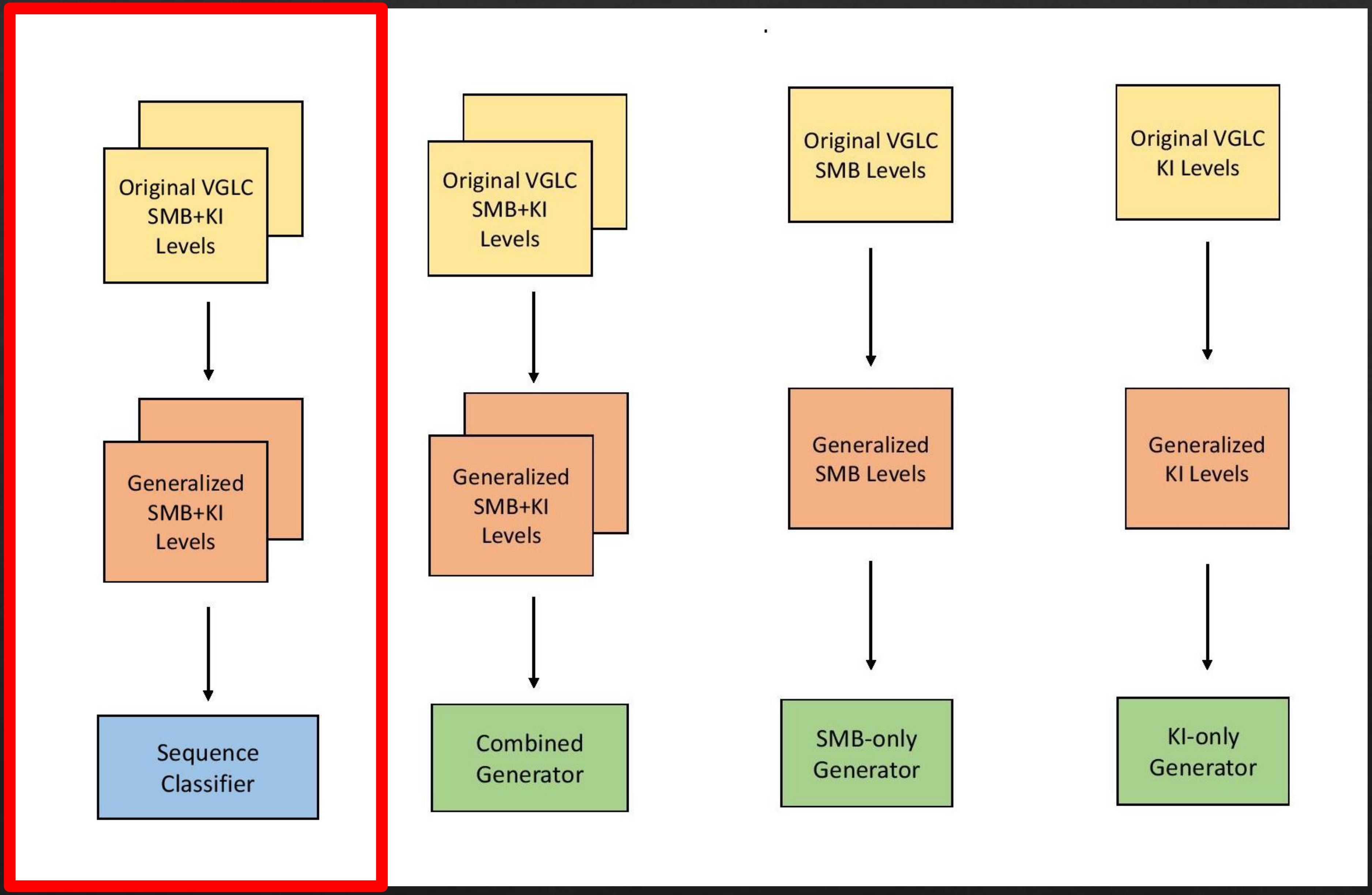
Original VGLC SMB+KI Levels
Generalized SMB+KI Levels
Combined Generator
Original VGLC SMB Levels
Generalized SMB Levels
SMB-only Generator
Original VGLC KI Levels
Generalized KI Levels
KI-only Generator

Models



Should a generated sequence be laid out like an SMB column or a KI row?

Models



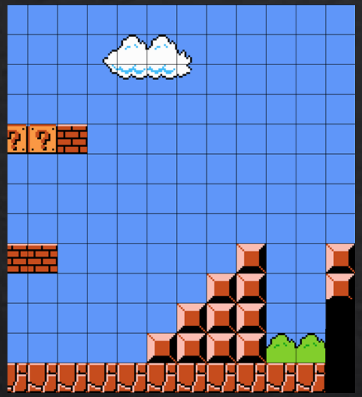
Original VGLC SMB+KI Levels
Generalized SMB+KI Levels
Sequence Classifier
Original VGLC SMB+KI Levels
Generalized SMB+KI Levels
Combined Generator
Original VGLC SMB Levels
Generalized SMB Levels
SMB-only Generator
Original VGLC KI Levels
Generalized KI Levels
KI-only Generator

Level Generation

- Forward an initial level seed through the LSTM which repeatedly predicts next most likely character given previous characters in the string until full level is generated

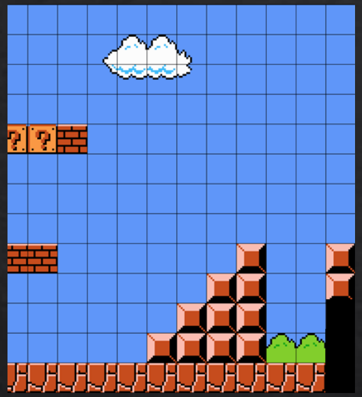
Level Generation

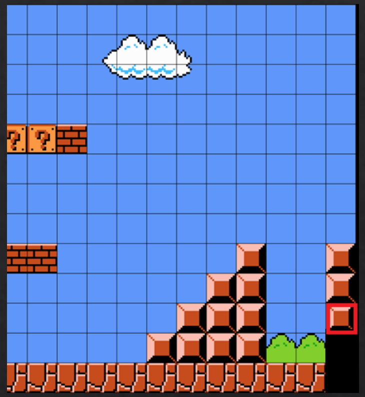
- Forward an initial level seed through the LSTM which repeatedly predicts next most likely character given previous characters in the string until full level is generated



Level Generation

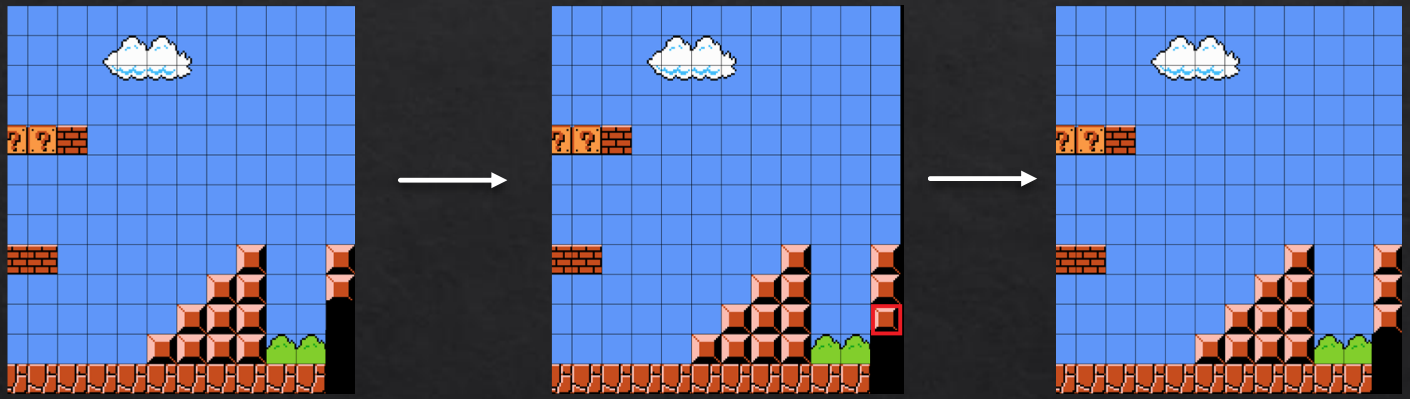
- Forward an initial level seed through the LSTM which repeatedly predicts next most likely character given previous characters in the string until full level is generated





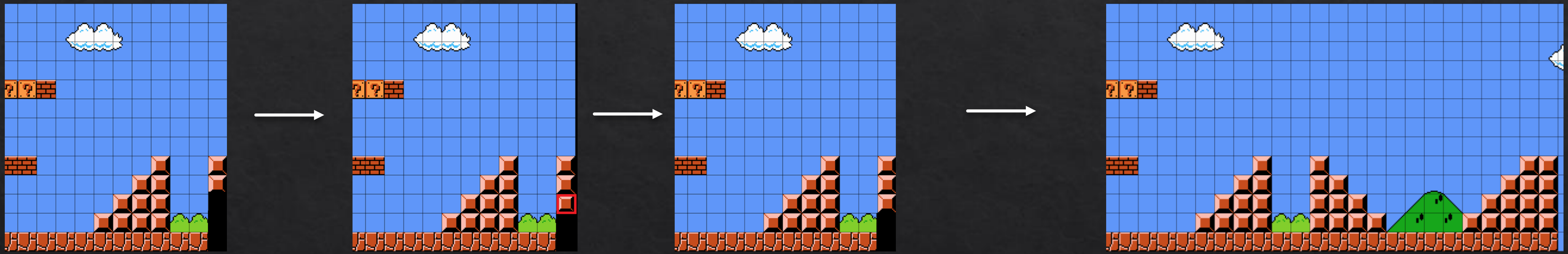
Level Generation

- Forward an initial level seed through the LSTM which repeatedly predicts next most likely character given previous characters in the string until full level is generated



Level Generation

- Forward an initial level seed through the LSTM which repeatedly predicts next most likely character given previous characters in the string until full level is generated



Credit: Adam Geitgey, Medium

Level Generation

- Three generators

Level Generation

- Three generators
  - Unweighted generator UW that used model trained on combined dataset

Level Generation

- Three generators
  - Unweighted generator UW that used model trained on combined dataset
  - Weighted generator WC that used model trained on combined dataset

Level Generation

- Three generators
  - Unweighted generator UW that used model trained on combined dataset
  - Weighted generator WC that used model trained on combined dataset
  - Weighted generator WS that used the models trained separately i.e. consisted of an SMB-only sub generator and a KI-only sub generator

Level Generation

- Three generators
  - Unweighted generator UW that used model trained on combined dataset
  - Weighted generator WC that used model trained on combined dataset
  - Weighted generator WS that used the models trained separately i.e. consisted of an SMB-only sub generator and a KI-only sub generator
- For UW, generated levels consisting of 200 sequences

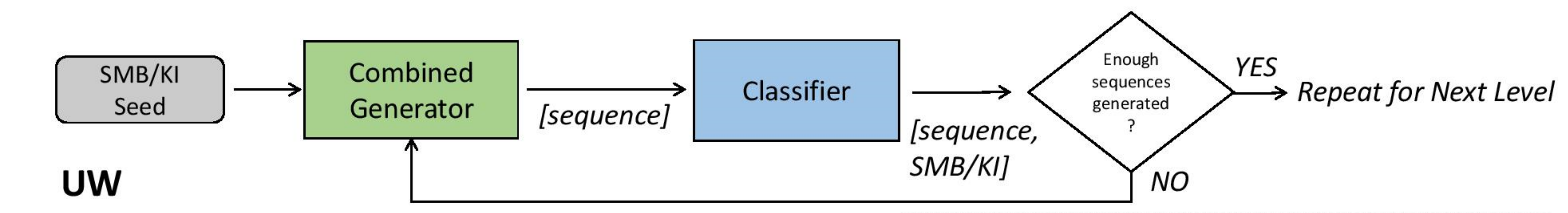
Level Generation

- Three generators
  - Unweighted generator UW that used model trained on combined dataset
  - Weighted generator WC that used model trained on combined dataset
  - Weighted generator WS that used the models trained separately i.e. consisted of an SMB-only sub generator and a KI-only sub generator

- For UW, generated levels consisting of 200 sequences

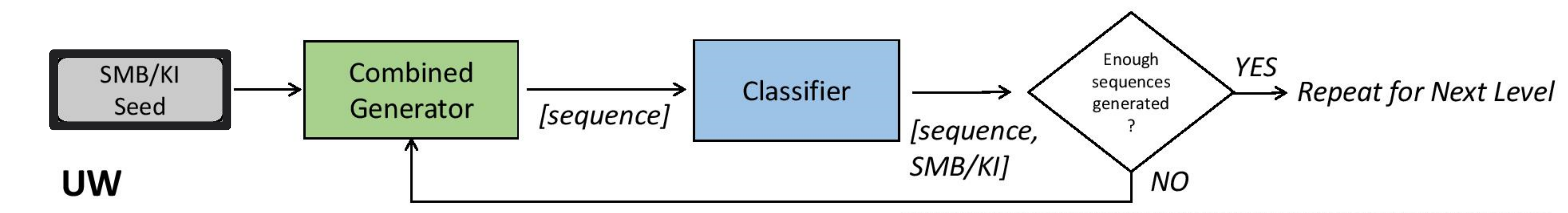
- For both WC and WS, generated levels consisting of 10 segments of 20 sequences each

Level Generation



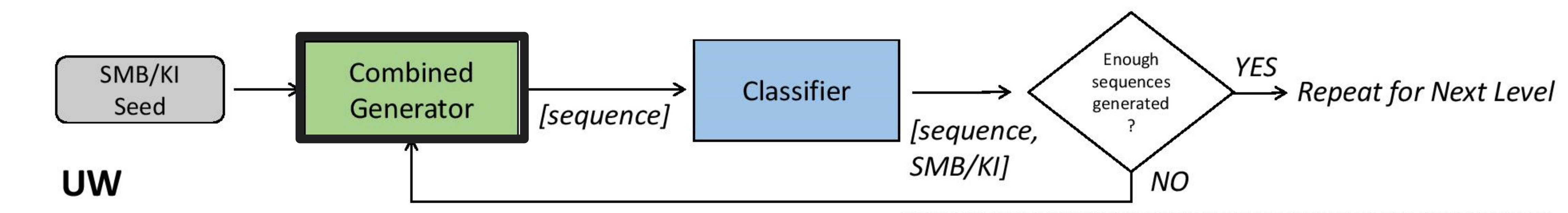
Unweighted Generator

Level Generation



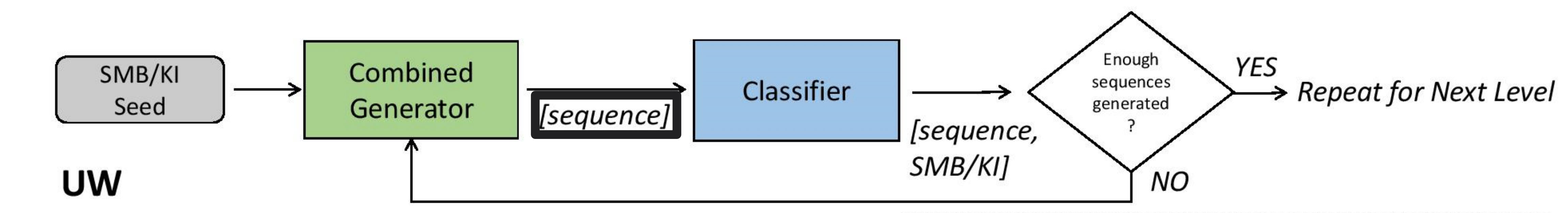
Unweighted Generator

Level Generation



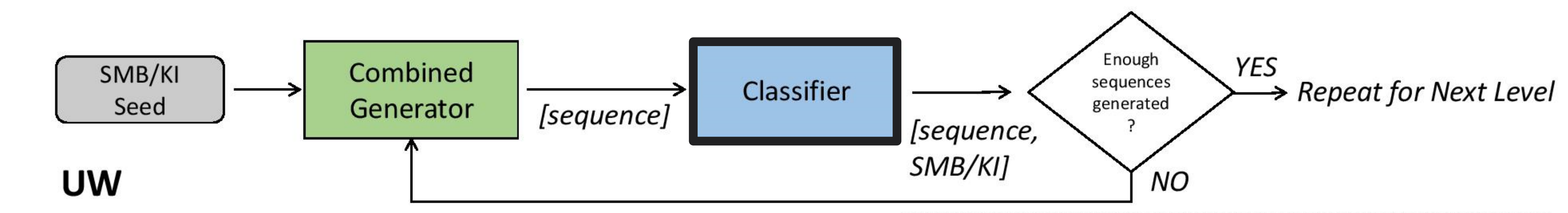
Unweighted Generator

Level Generation



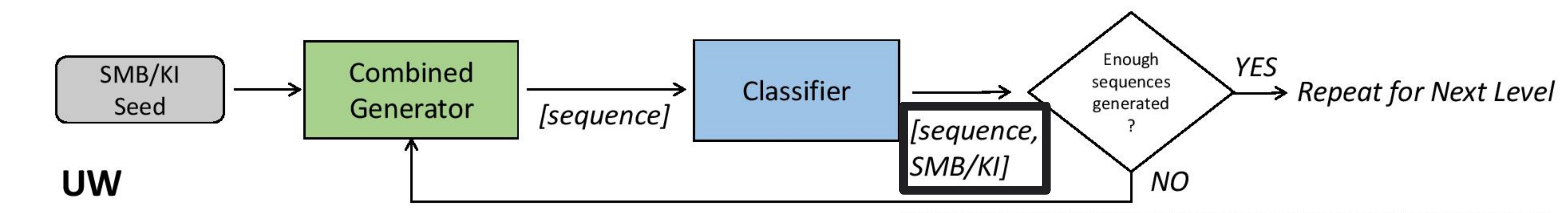
Unweighted Generator

Level Generation



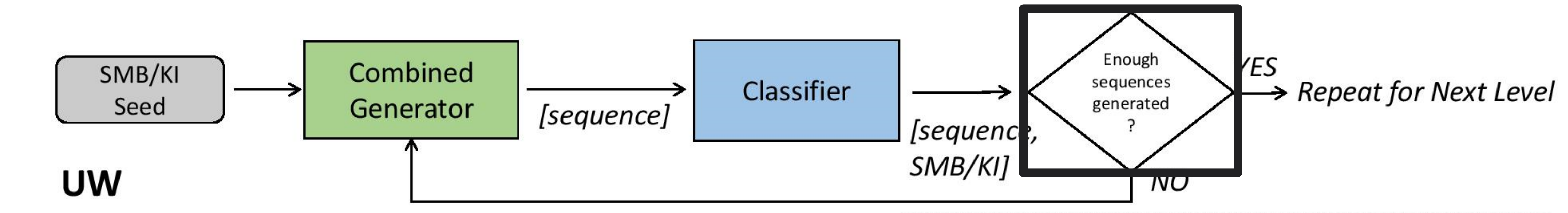
Unweighted Generator

Level Generation



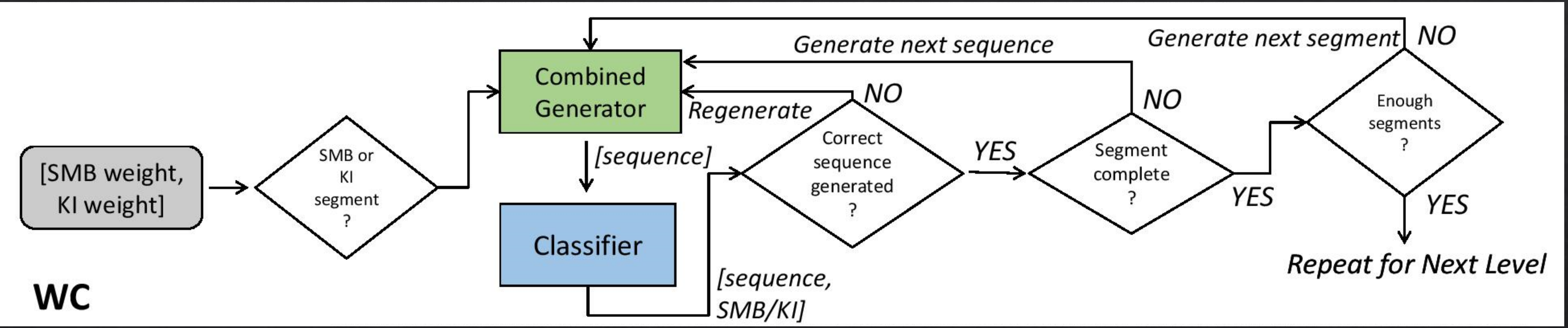
Unweighted Generator

Level Generation



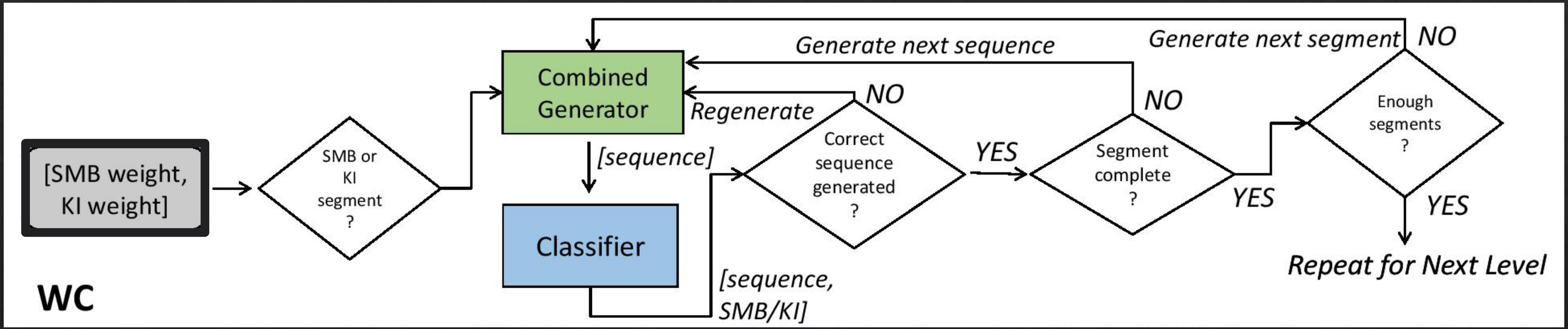
Unweighted Generator

Level Generation



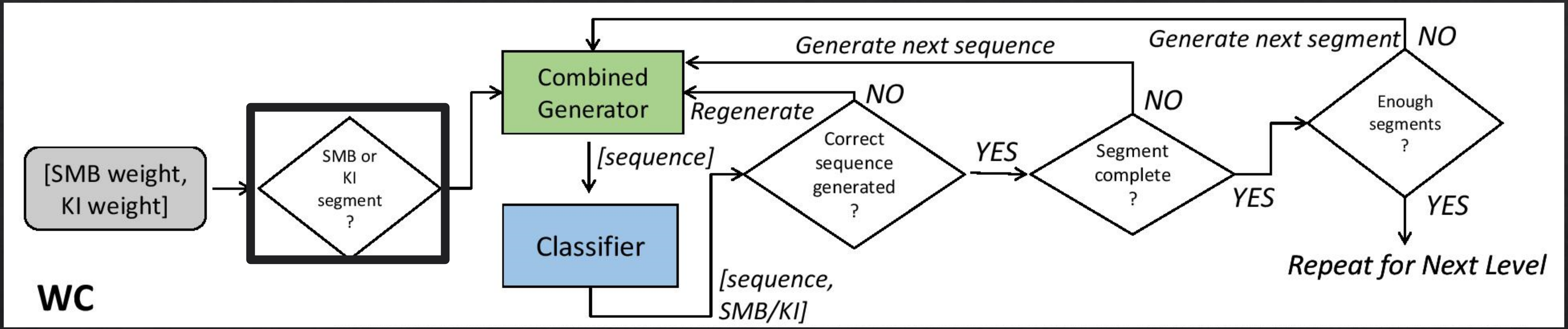
Weighted Generator 1

Level Generation



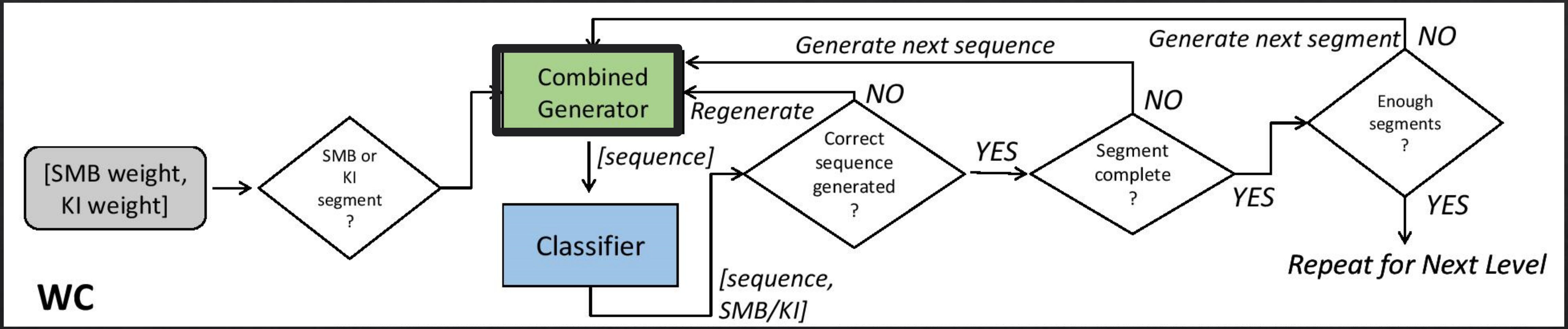
Weighted Generator 1

Level Generation



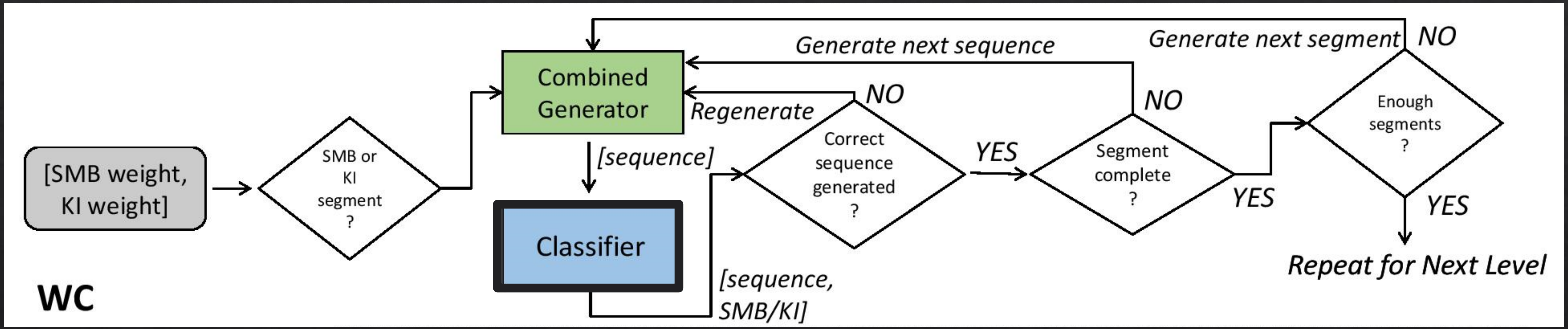
Weighted Generator 1

Level Generation



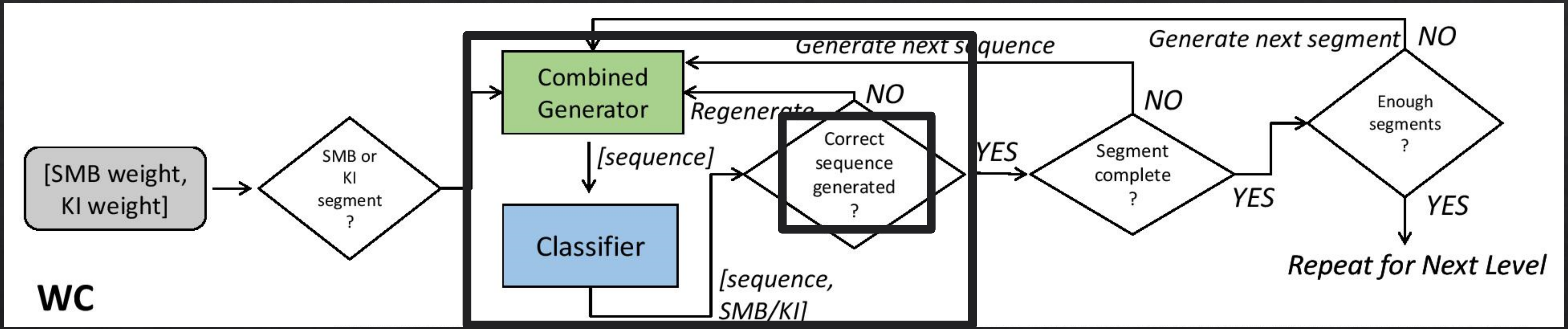
Weighted Generator 1

Level Generation



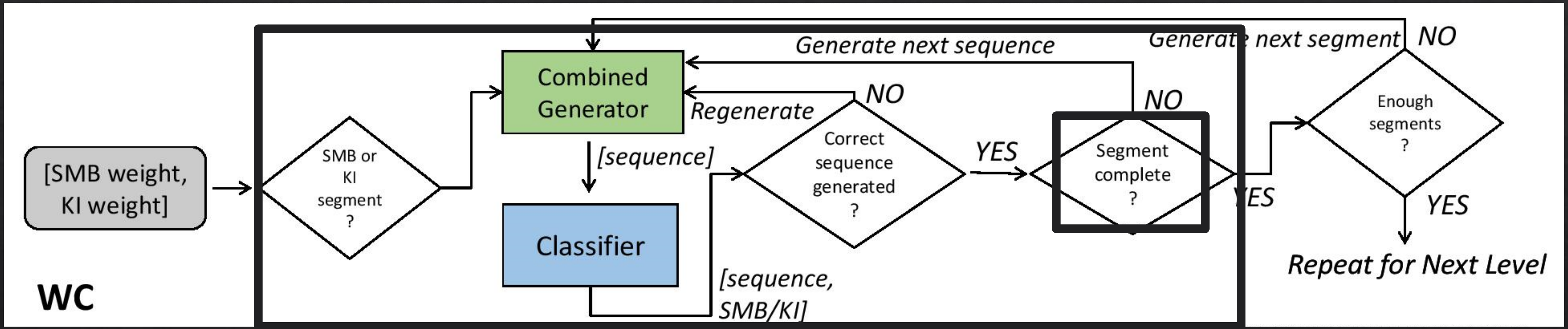
Weighted Generator 1

Level Generation



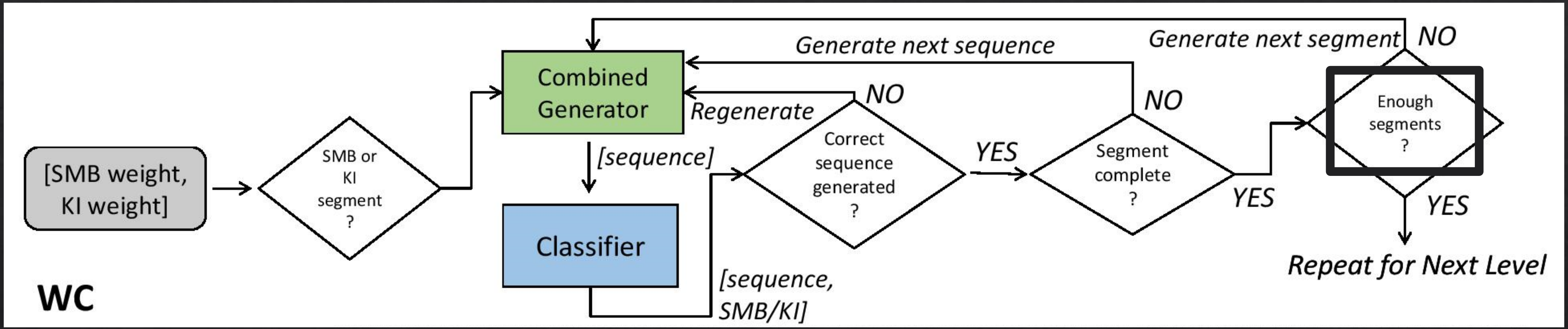
Weighted Generator 1

Level Generation



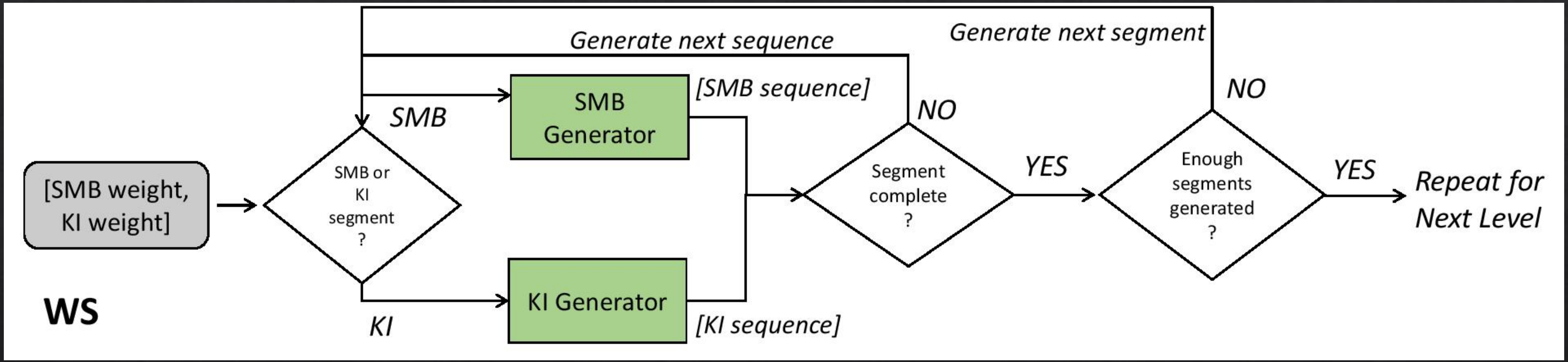
Weighted Generator 1

Level Generation



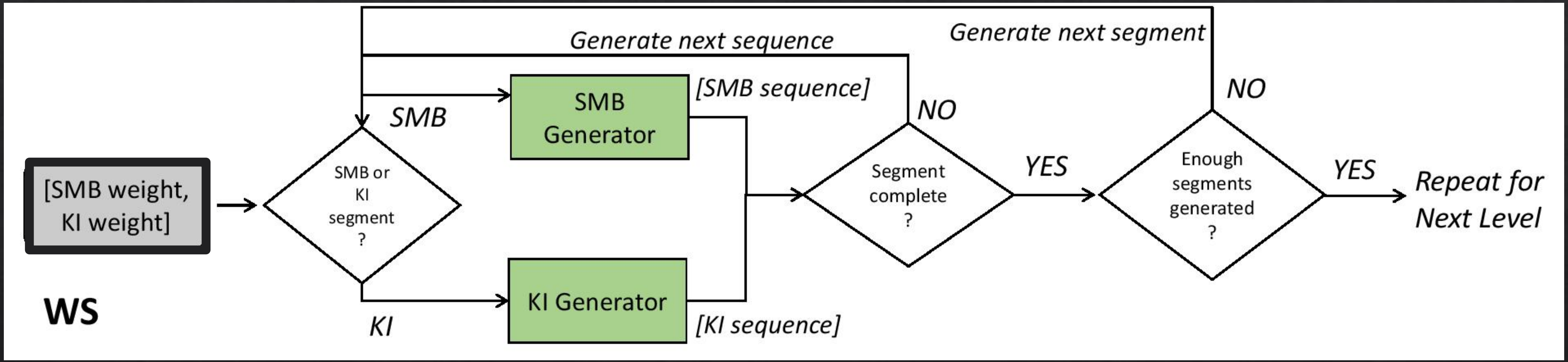
Weighted Generator 1

Level Generation



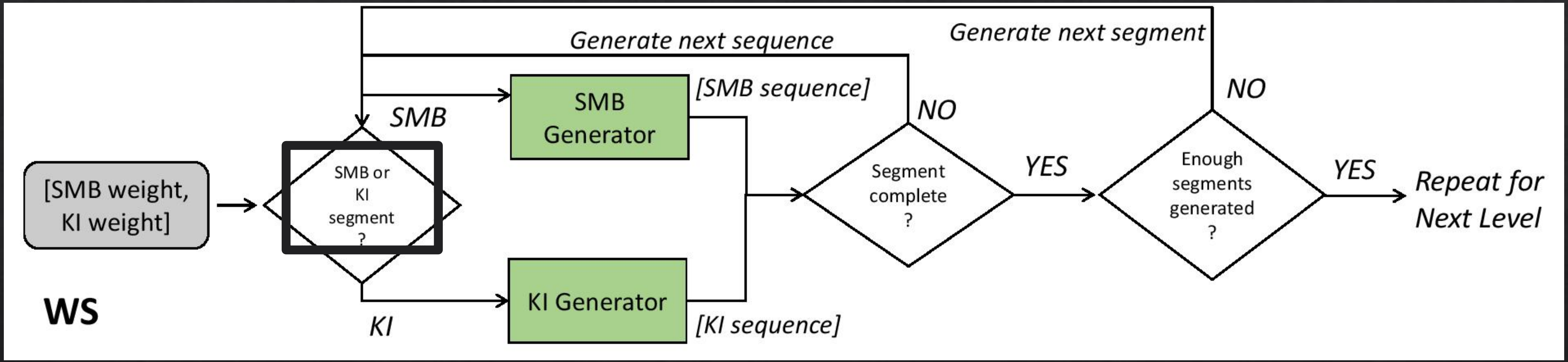
Weighted Generator 2

Level Generation



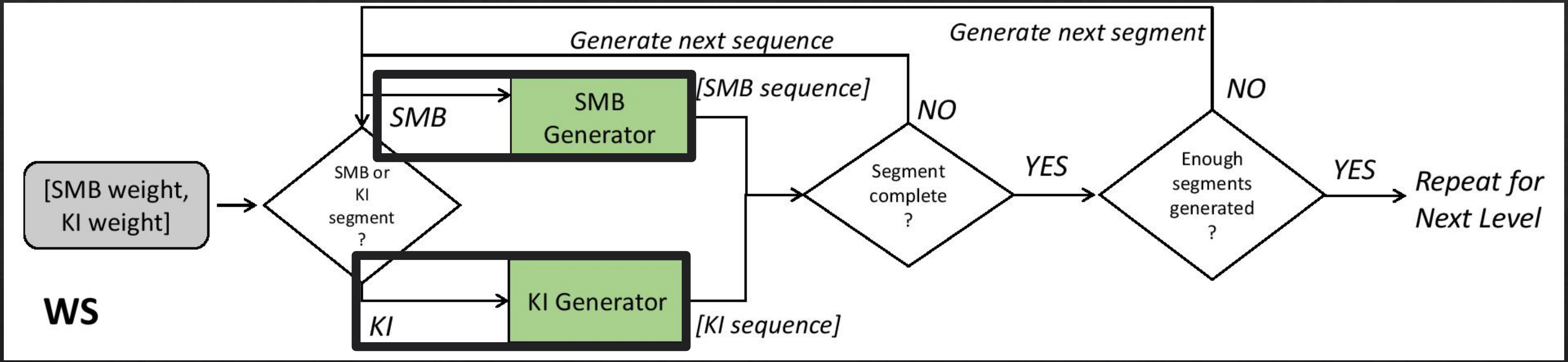
Weighted Generator 2

Level Generation



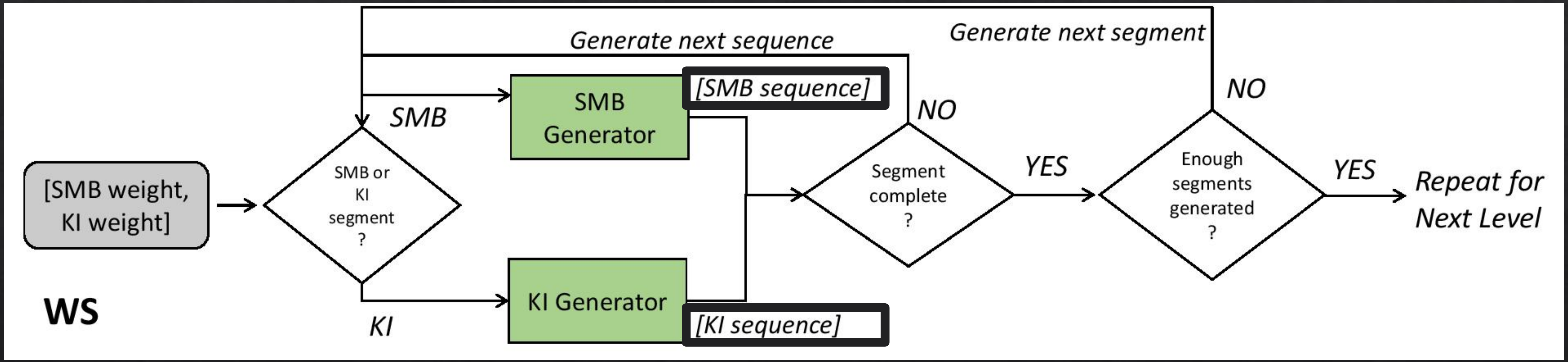
Weighted Generator 2

Level Generation



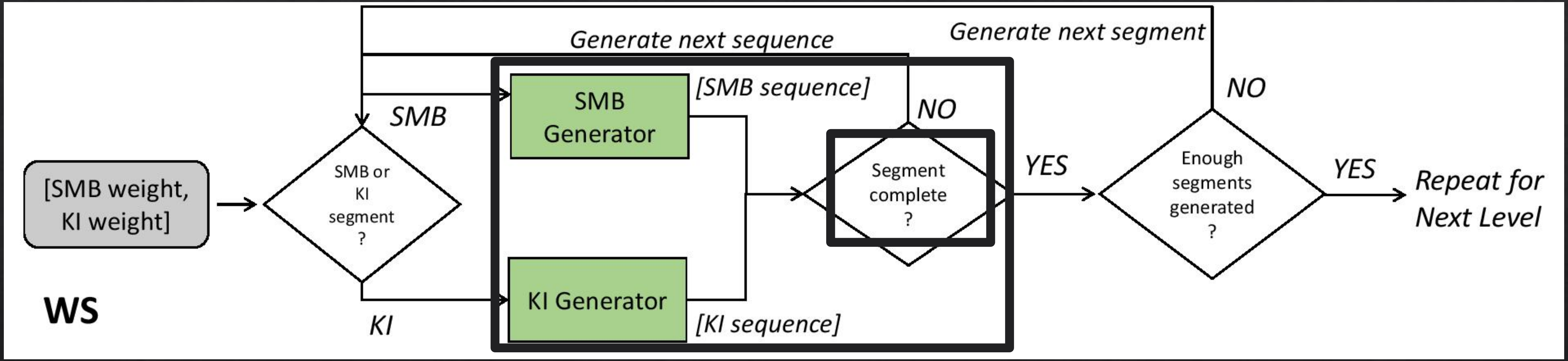
Weighted Generator 2

Level Generation



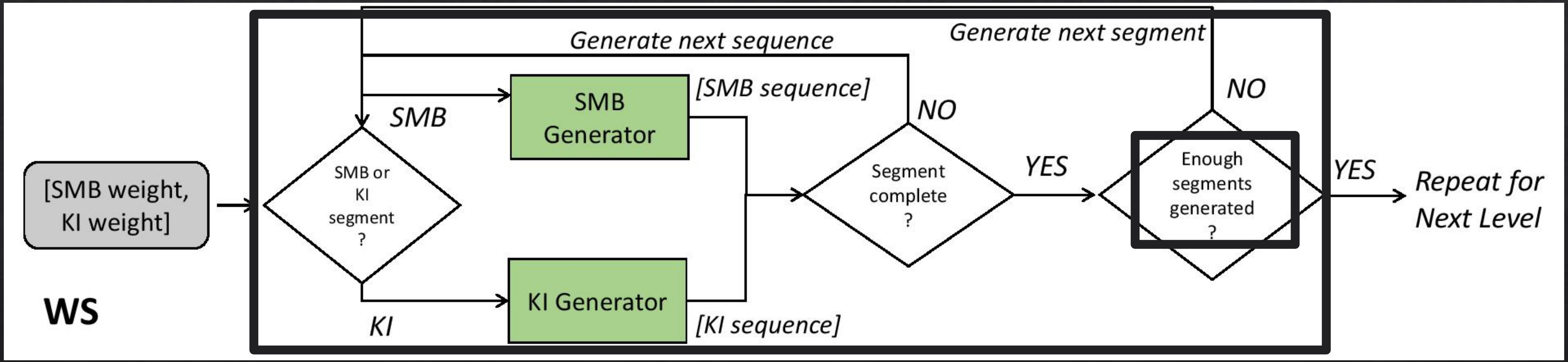
Weighted Generator 2

Level Generation



Weighted Generator 2

Level Generation



Weighted Generator 2

Layout

- Generated sequences are laid out using a basic algorithm

Layout

- Generated sequences are laid out using a basic algorithm

- Three cases:
  - Column after column/Row after row
  - Row after column
  - Column after row

Layout

- Generated sequences are laid out using a basic algorithm

- Three cases:
  - Column after column/Row after row
    - Stack one after another
  - Row after column
  - Column after row

Layout

- Generated sequences are laid out using a basic algorithm
- Three cases:
  - Column after column/Row after row
  - Row after column
    - Align row with topmost point of column on which player can stand
  - Column after row

Layout

- Generated sequences are laid out using a basic algorithm

- Three cases:
  - Column after column/Row after row
  - Row after column
  - Column after row
    - Align topmost point of column on which player can stand with the row

Layout

- Generated sequences are laid out using a basic algorithm

- Three cases:
  - Column after column/Row after row
  - Row after column
  - Column after row

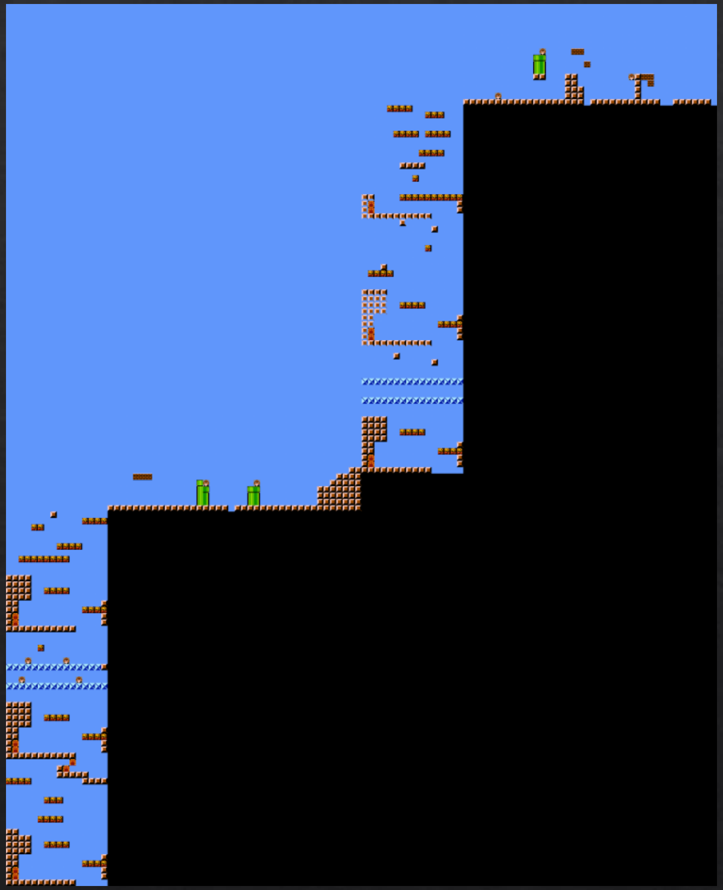
- Layout function separate from generation

Example Levels

Unweighted Generation

Example Levels

WC

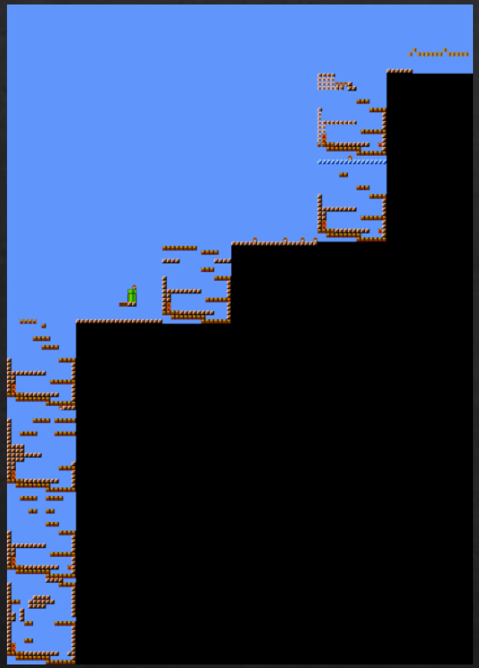


WS

Weighted Generation (0.5, 0.5)

Weighted Generation

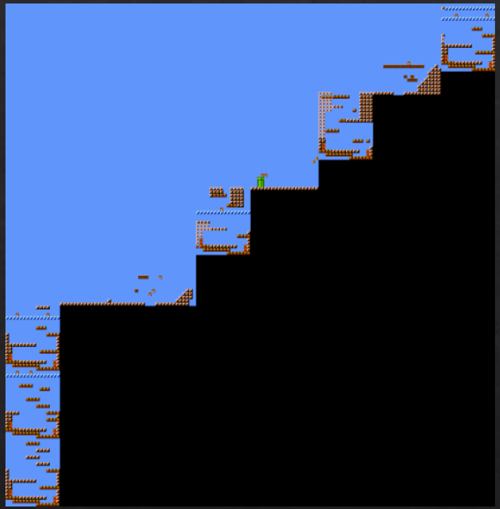
Weighted Generation



(SMB=0.2, KI=0.8)

Weighted Generation

(SMB=0.2, KI=0.8)

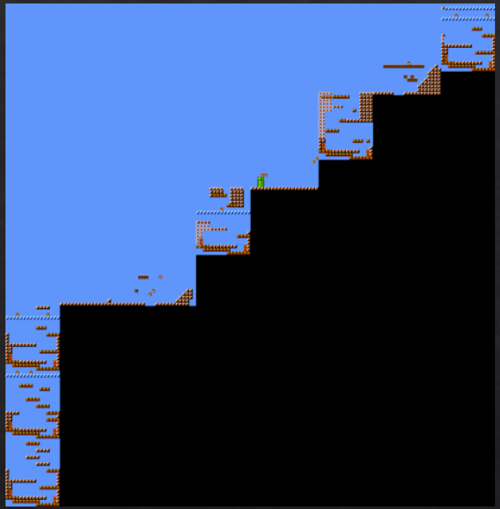


(SMB=0.4, KI=0.6)

Weighted Generation

(SMB=0.5, KI=0.5)

(SMB=0.2, KI=0.8)

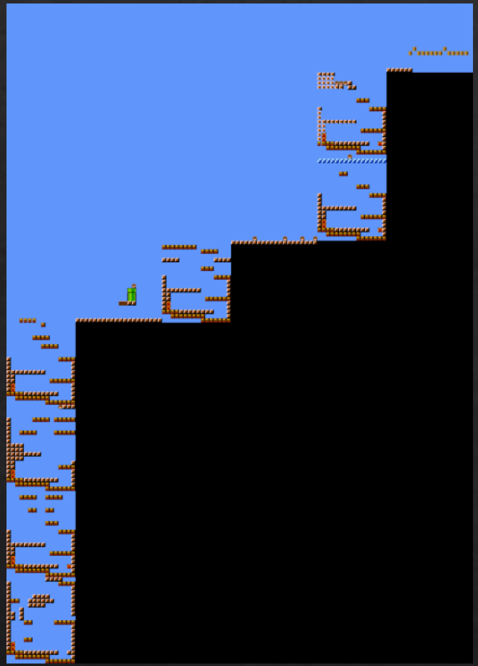


(SMB=0.4, KI=0.6)

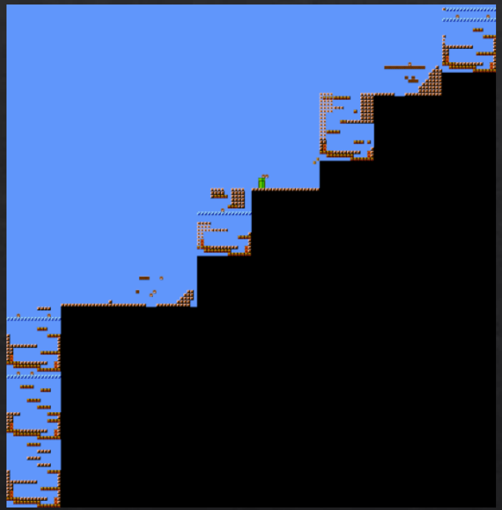
Weighted Generation

(SMB=0.5, KI=0.5)

(SMB=0.6, KI=0.4)



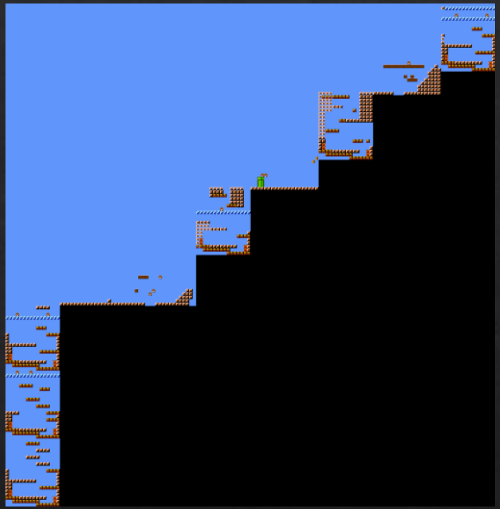
(SMB=0.2, KI=0.8)



(SMB=0.4, KI=0.6)

Weighted Generation

(SMB=0.2, KI=0.8)



(SMB=0.4, KI=0.6)

(SMB=0.5, KI=0.5)

(SMB=0.6, KI=0.4)

(SMB=0.8, KI=0.2)

Measures

Measures

$$Leniency = \frac{-(\#\ Enemy\ Sprites + (0.5 \ast \#\ Gaps))}{\#\ Sequences\ in\ Level}$$

Measures

$$Leniency$$

$$\frac{-(\#\ Enemy\ Sprites + (0.5\ * \#\ Gaps))}{\#\ Sequences\ in\ Level}$$

$$Density$$

$$\frac{(\#\ Ground + \#\ Platform)}{\#\ Sequences\ in\ Level}$$

Measures

$$Leniency$$
$$\frac{-(\#\ Enemy\ Sprites + (0.5\ *\#\ Gaps))}{\#\ Sequences\ in\ Level}$$

$$Density$$
$$\frac{(\#\ Ground + \#\ Platform)}{\#\ Sequences\ in\ Level}$$

$$Sequence\ Density$$
$$\frac{\#\ Sequences\ in\ level\ in\ training\ set}{\#\ Sequences\ in\ Level}$$

$$Sequence\ Variation$$
$$\frac{\#\ Unique\ Sequences\ in\ level\ in\ training\ set}{\#\ Sequences\ in\ Level}$$

Measures

$$Leniency = \frac{-(\#\ Enemy\ Sprites + (0.5\ * \#\ Gaps))}{\#\ Sequences\ in\ Level}$$

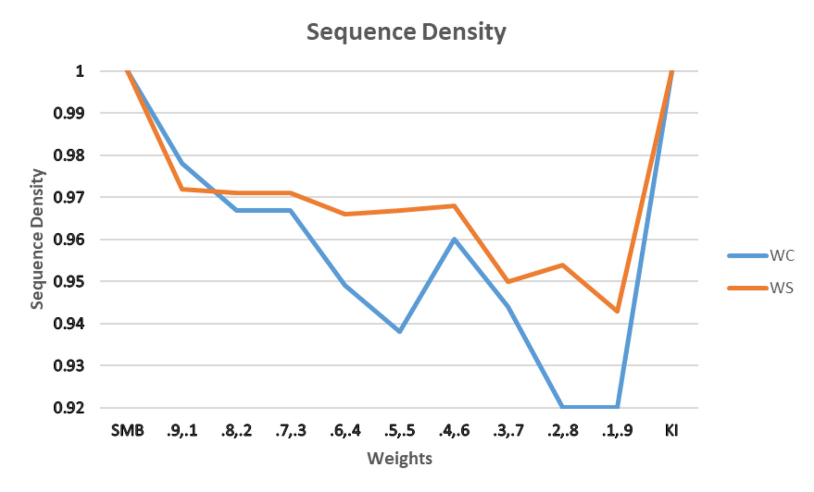
$$Density = \frac{(\#\ Ground + \#\ Platform)}{\#\ Sequences\ in\ Level}$$

$$Sequence\ Density = \frac{\#\ Sequences\ in\ level\ in\ training\ set}{\#\ Sequences\ in\ Level}$$

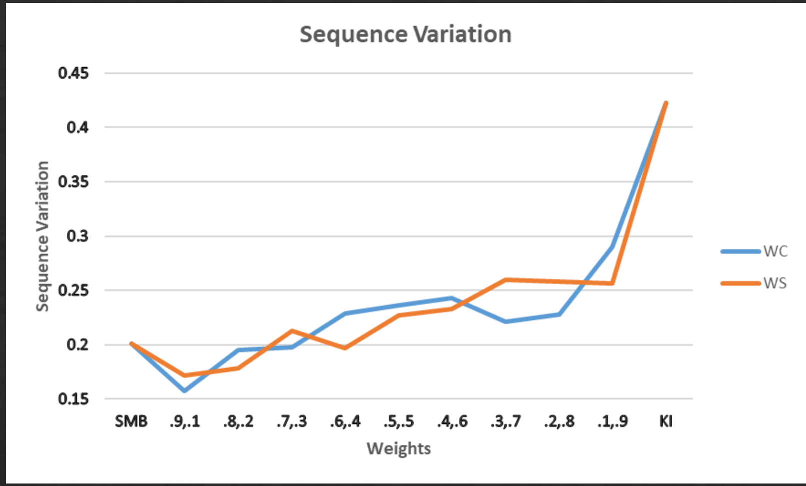
$$Sequence\ Variation = \frac{\#\ Unique\ Sequences\ in\ level\ in\ training\ set}{\#\ Sequences\ in\ Level}$$

$$Aspect\ Ratio = \frac{\#\ Rows\ in\ Level}{\#\ Columns\ in\ Level}$$

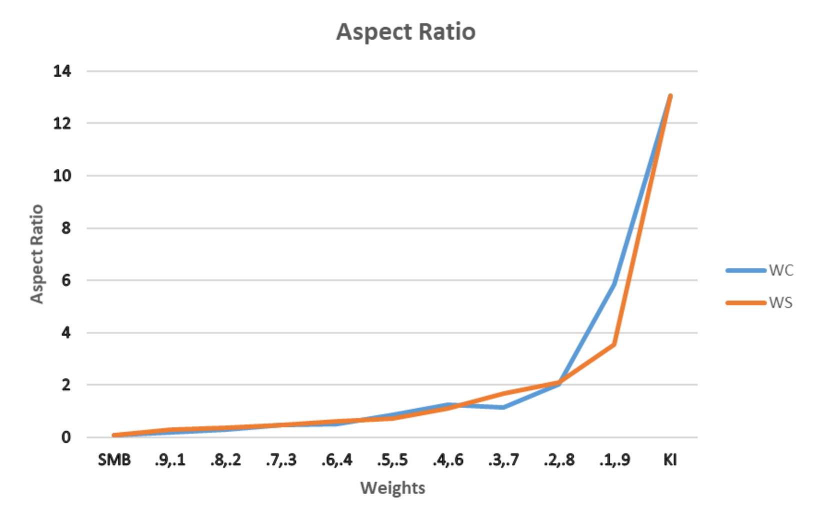
Results



Sequence Density

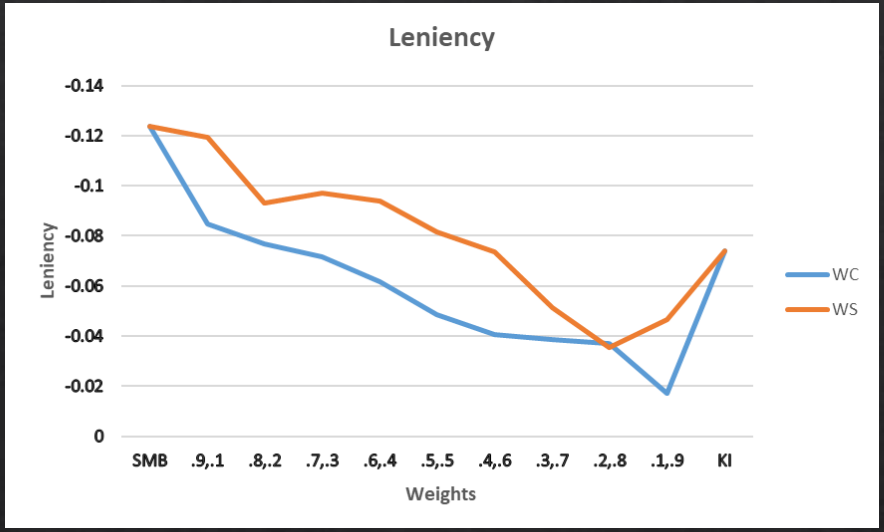


Sequence Variation

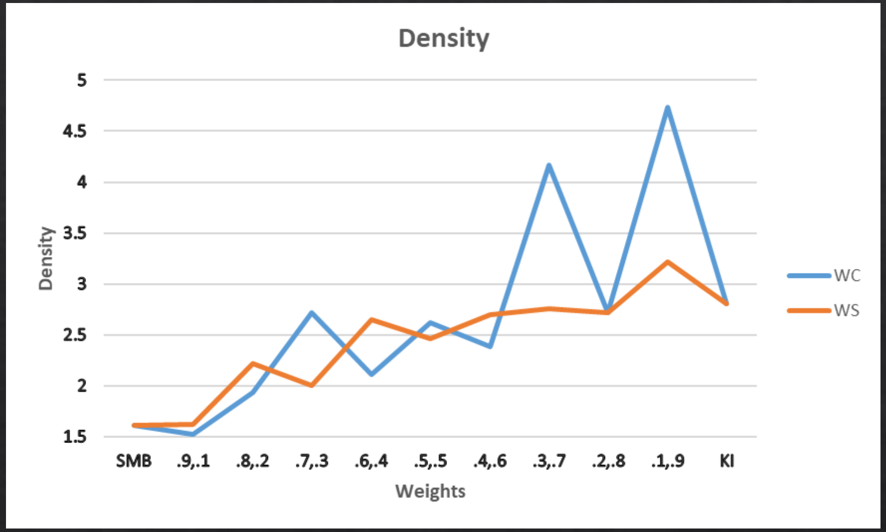


Aspect Ratio

Results



Leniency



Density

Discussion

- Altering weights impacted the type of levels generated and roughly interpolated between SMB and KI

Discussion

- Altering weights impacted the type of levels generated and roughly interpolated between SMB and KI

- Possible to generate levels that are a mix of levels from 2 games but can also be made to be more like one than the other

Discussion

- Altering weights impacted the type of levels generated and roughly interpolated between SMB and KI
- Possible to generate levels that are a mix of levels from 2 games but can also be made to be more like one than the other
- Deviations suggest that these methods can also produce some novelty

Future Work

Future Work

- No playability tests were run nor playability/path-based information used in training, thus levels are currently not completable; using an agent to carve-out a path post-generation or encoding path info into training corpus could help

Future Work

- No playability tests were run nor playability/path-based information used in training, thus levels are currently not completable; using an agent to carve-out a path post-generation or encoding path info into training corpus could help

- Blended levels necessitate blended mechanics to be fully playable

Future Work

- No playability tests were run nor playability/path-based information used in training, thus levels are currently not completable; using an agent to carve-out a path post-generation or encoding path info into training corpus could help
- Blended levels necessitate blended mechanics to be fully playable
- Other techniques such as evolutionary algorithms to evolve game mechanics

Contact

Anurag Sarkar
Northeastern University
sarkar.an@husky.neu.edu