Neuro-Symbolic Modeling: Overview

Some examples of neural-symbolic integration

THE UNIVERSITY OF UTAH

This lecture

- The Two Systems of Thinking
- Learning & Reasoning
- History: Statistical relation learning
- Some examples of neural-symbolic integration
- Technical challenges for neural-symbolic integration
- A taxonomy of approaches

This lecture

- The Two Systems of Thinking
- Learning & Reasoning
- History: Statistical relation learning
- Some examples of neural-symbolic integration
- Technical challenges for neural-symbolic integration
- A taxonomy of approaches

Let's look at a few examples

1. Recognizing digits and adding them

2. Semantic role labeling

3. Using the rules of a game to play the game

Let’s look at a few examples

1. Recognizing digits and adding them
2. Semantic role labeling
3. Using the rules of a game to play the game

Digit prediction

Examples from the MNIST digit recognition dataset

The standard task

Given an image, recognize the digit in the image

Typically modeled as a multiclass classification task

Convolutional neural networks are near perfect

Digit prediction

Examples from the MNIST digit recognition dataset

The standard task

Given an image, recognize the digit in the image

Typically modeled as a multiclass classification task

Convolutional neural networks are near perfect

Label() = 5   Label( ) = 1

Digit prediction

Examples from the MNIST digit recognition dataset

The standard task

Given an image, recognize the digit in the image

Typically modeled as a multiclass classification task

Convolutional neural networks are near perfect

Label() = 5 Label( ) = 1

↓

The label for the input  is 5

Digit prediction

Examples from the MNIST digit recognition dataset

The standard task

Given an image, recognize the digit in the image

Typically modeled as a multiclass classification task

Convolutional neural networks are near perfect

Label() = 5

The label for the input  is 5

Label() = 1

The label for the input is 1

Digit prediction

Examples from the MNIST digit recognition dataset

The standard task

Given an image, recognize the digit in the image

Typically modeled as a multiclass classification task

Convolutional neural networks are near perfect

Label() = 5

Label( ) = 1

The label for the input  is 5

The label for the input is 1

These statements can be written as predicates

`Digit(`  `, 5)`

`Digit(` `, 1)`

Model decisions as predicates

“The model predicts a digit y for an input image x” = `Digit`(x, y)

Model decisions as predicates

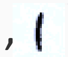
“The model predicts a digit y for an input image x” = `Digit`(x, y)

We can imagine another model that predicts the sum of two digit images

Model decisions as predicates

"The model predicts a digit y for an input image x" = `Digit` $(x,\ y)$

We can imagine another model that predicts the sum of two digit images

Sum( , ) = 6 The sum of these two digits is 6

Model decisions as predicates

"The model predicts a digit y for an input image x" = `Digit` $(x,\ y)$

We can imagine another model that predicts the sum of two digit images

Sum( ,) = 6 The sum of these two digits is 6

As a predicate: `Sum` ( , , 6)

Model decisions as predicates

"The model predicts a digit y for an input image x" = `Digit` $(x,\ y)$

We can imagine another model that predicts the sum of two digit images

Sum( ,) = 6 The sum of these two digits is 6

As a predicate: `Sum` ( , , 6)

"The model predicts a sum y for input images x_1, x_2 " = `Sum` $(x_1,\ x_2,\ y)$

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1, z_1) \land \texttt{Digit}(x_2, z_2) \land (y = z_1 + z_2) \rightarrow \texttt{Sum}(x_1, x_2, y)$$

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\boxed{\texttt{Digit}(x_1,\ z_1)} \land \texttt{Digit}(x_2,\ z_2) \land (y = z_1 + z_2) \rightarrow \texttt{Sum}(x_1, x_2,\ y)$$

`If` The image x_1
has the digit z_1

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1,\ z_1) \land \boxed{\texttt{Digit}(x_2,\ z_2)} \land (y = z_1 + z_2) \rightarrow \texttt{Sum}(x_1, x_2,\ y)$$

`If` The image x_1 has the digit z_1 `and` The image x_2 has the digit z_2

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1,\ z_1) \wedge \texttt{Digit}(x_2,\ z_2) \wedge \boxed{(y = z_1 + z_2)} \rightarrow \texttt{Sum}(x_1,\ x_2,\ y)$$

`If` The image x_1 has the digit z_1 `and` The image x_2 has the digit z_2 `and` The numbers z_1 and z_2 add up to y

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1,\ z_1) \land \texttt{Digit}(x_2,\ z_2) \land (y = z_1 + z_2) \rightarrow \boxed{\texttt{Sum}(x_1,\ x_2,\ y)}$$

`If`	The image x_1 has the digit z_1	`and`	The image x_2 has the digit z_2	`and`	The numbers z_1 and z_2 add up to y	`then`	The label for the sum model when given the images x_1 and x_2 should be y

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1,\ z_1) \wedge \texttt{Digit}(x_2,\ z_2) \wedge (y = z_1 + z_2) \rightarrow \texttt{Sum}(x_1,\ x_2,\ y)$$

`If`	The image x_1 has the digit z_1	`and`	The image x_2 has the digit z_2	`and`	The numbers z_1 and z_2 add up to y	`then`	The label for the sum model when given the images x_1 and x_2 should be y

Digits and sums

The `Digit` and `Sum` models can be both implemented as CNNs

But their predictions are not independent of each other

$$\texttt{Digit}(x_1,\ z_1) \wedge \texttt{Digit}(x_2,\ z_2) \wedge (y = z_1 + z_2) \rightarrow \texttt{Sum}(x_1, x_2,\ y)$$

Given data for one task and this rule, can we train a model for the other task?

Let’s look at a few examples

1. Recognizing digits and adding them

2. Semantic role labeling

3. Using the rules of a game to play the game

Semantic Role Labeling (SRL)

Who did what to whom, where, when, why?

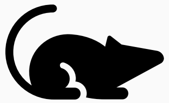
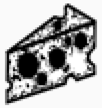
The mouse ate the cheese when the farmer left the kitchen.

Semantic Role Labeling (SRL)

Who did what to whom, where, when, why?

The mouse ate the cheese when the farmer left the kitchen.

Arg0 Arg1 ArgM-TMP

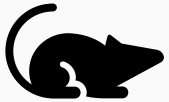
	ate
Arg0	The mouse
Arg1	the cheese
ArgM-TMP	when the farmer left the kitchen

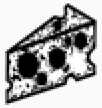
Semantic Role Labeling (SRL)

Who did what to whom, where, when, why?

The mouse ate the cheese when the farmer left the kitchen.

the farmer	left	the kitchen
Arg0		Arg1





	ate
Arg0	The mouse
Arg1	the cheese
ArgM-TMP	when the farmer left the kitchen

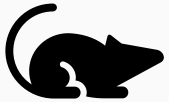
	left
Arg0	the farmer
Arg1	the kitchen

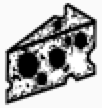
Semantic Role Labeling (SRL)

Who did what to whom, where, when, why?

The mouse ate the cheese when the farmer left the kitchen.

These *semantic roles* are defined by the PropBank data (Palmer et al)





	ate
Arg0	The mouse
Arg1	the cheese
ArgM-TMP	when the farmer left the kitchen

	left
Arg0	the farmer
Arg1	the kitchen

What do the labels mean?

The mouse ate the cheese when the farmer left the kitchen.

ate		left	
Arg0	The mouse	**Arg0**	the farmer
Arg1	the cheese	**Arg1**	the kitchen
ArgM-TMP	when the farmer left the kitchen		

PropBank defines what the labels `Arg0`, `Arg0` etc mean for each verb

What do the labels mean?

The mouse ate the cheese when the farmer left the kitchen.

Agent: eater, consumer

Patient: the meal

Temporal *modifier*

ate	
`Arg0`	The mouse
`Arg1`	the cheese
`ArgM-TMP`	when the farmer left the kitchen

left	
`Arg0`	the farmer
`Arg1`	the kitchen

PropBank defines what the labels `Arg0`, `Arg0` etc mean for each verb

What do the labels mean?

The mouse ate the cheese when the farmer left the kitchen.

Agent: eater, consumer

Patient: the meal

Temporal *modifier*

	ate
`Arg0`	The mouse
`Arg1`	the cheese
`ArgM-TMP`	when the farmer left the kitchen

	left
`Arg0`	the farmer
`Arg1`	the kitchen

Agent: entity in motion

Starting point, location vacated

PropBank defines what the labels `Arg0`, `Arg0` etc mean for each verb

Semantic Role Labeling: The modeling problem

- Input: A sentence
- Output: Semantic frames for all verbs

A well studied task, large datasets in English

- Penn Treebank data, Ontonotes annotated with PropBank roles
- Creating datasets is not easy though

Semantic Role Labeling: The modeling problem

- **Input**: A sentence
- **Output**: Semantic frames for all verbs

The output is *structured*, and has constraints about the labels. For example

- Core arguments (e.g. `Arg0`, `Arg1`) cannot repeat…
  …but modifiers (e.g. `ArgM-TMP`) can

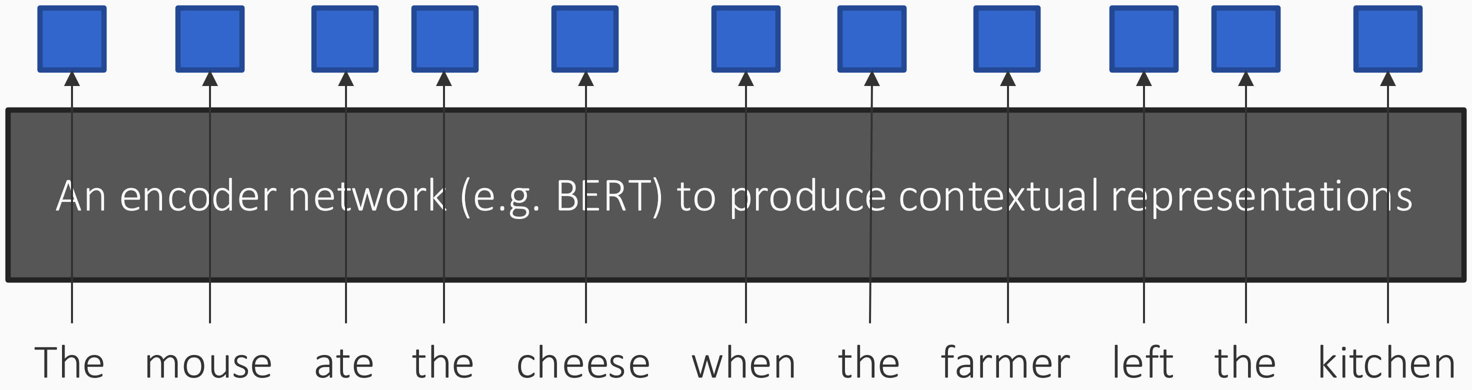
- Certain arguments (called references, e.g. `R-Arg0`) can appear only if the corresponding referent argument exists (here, `Arg0`)

These symbolic constraints come from the task definition. And linguistic assumptions.
We will see other constraints later

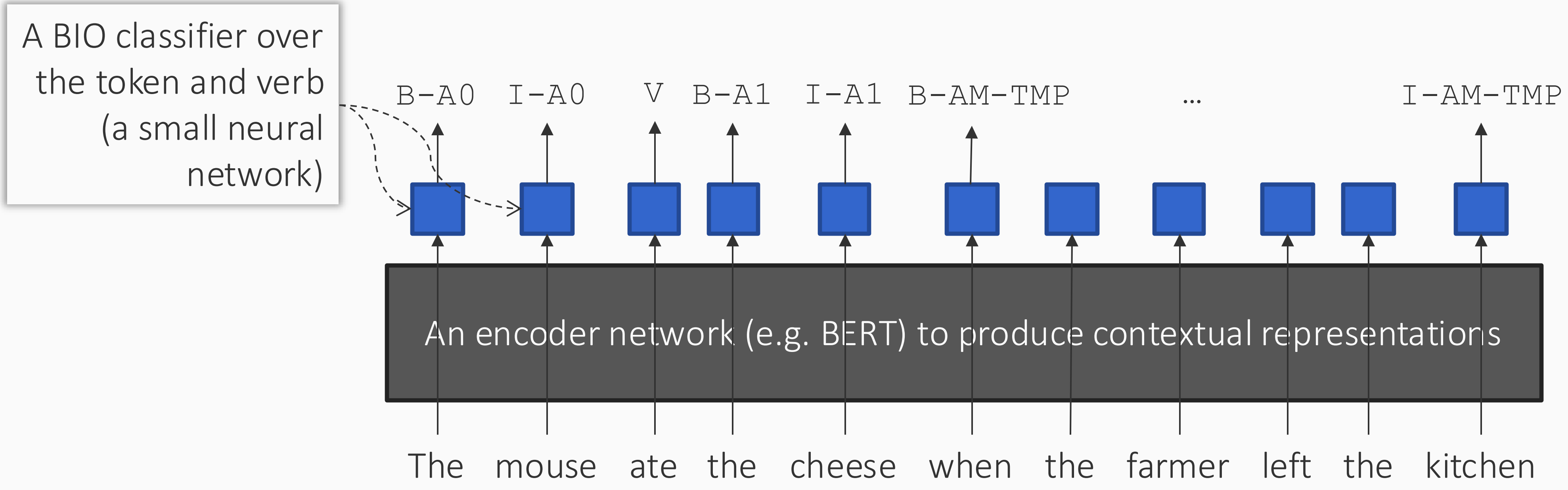
A common design of neural SRL models

The mouse ate the cheese when the farmer left the kitchen

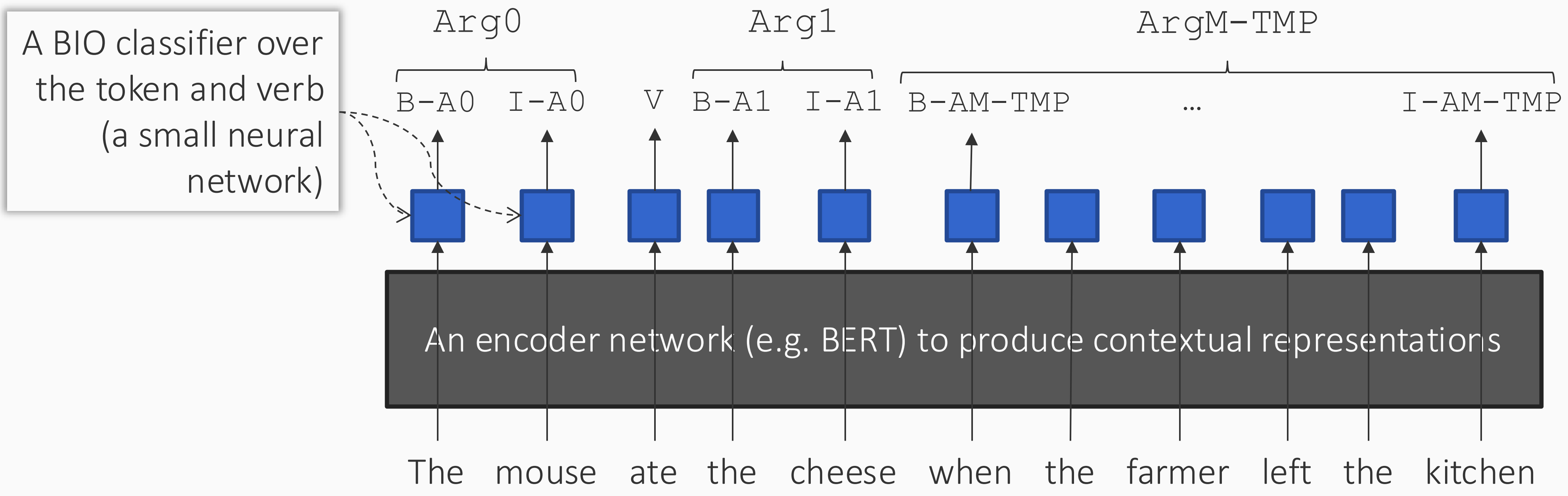
A common design of neural SRL models



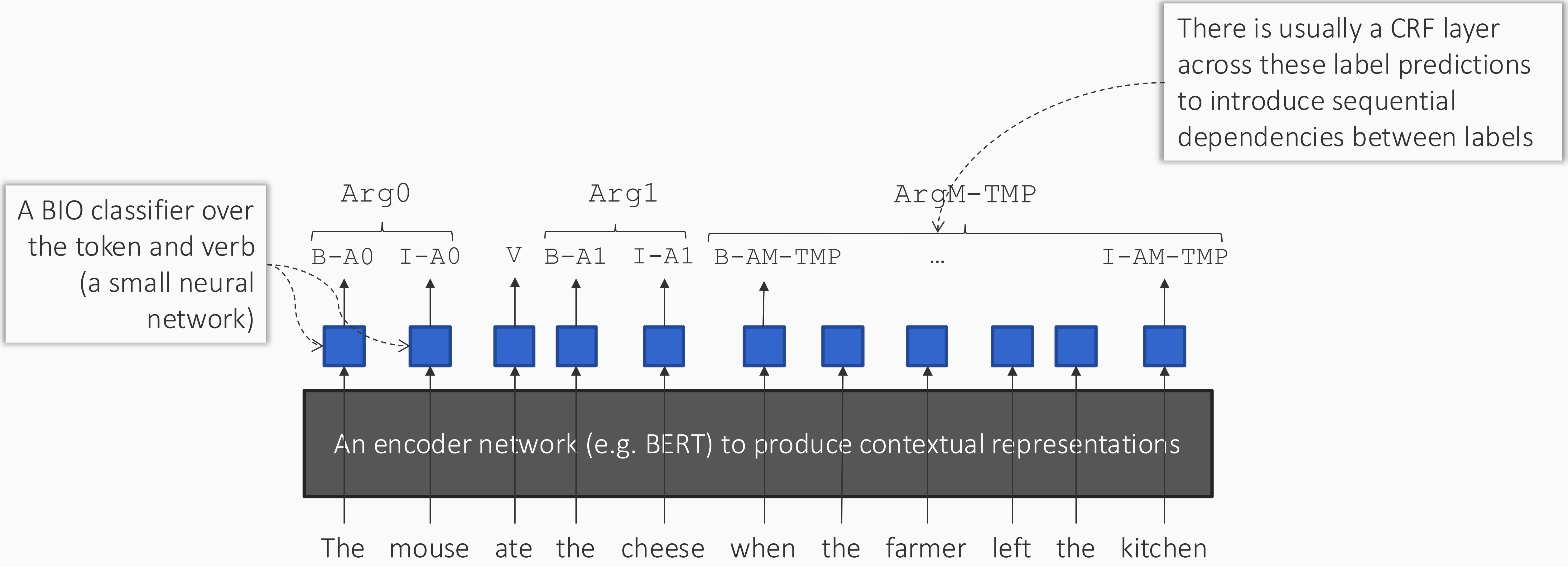
A common design of neural SRL models



A common design of neural SRL models

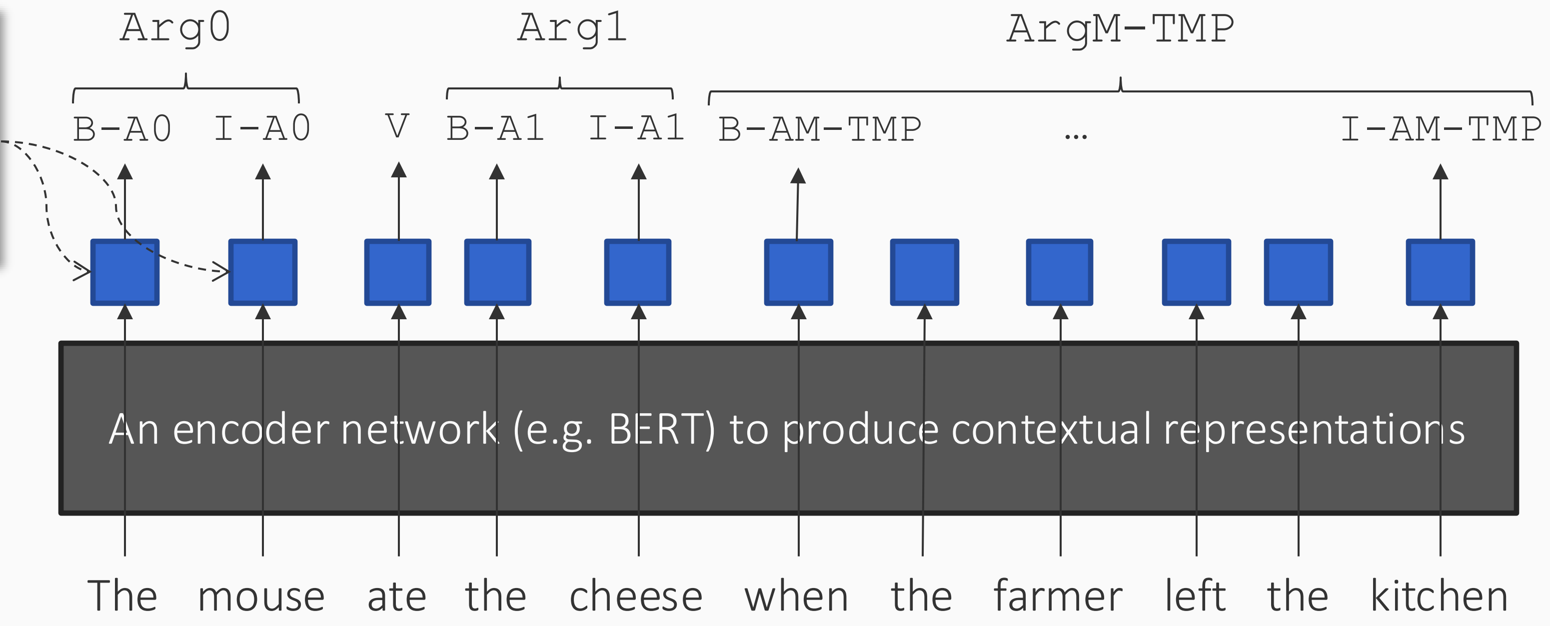


A common design of neural SRL models



A common design of neural SRL models

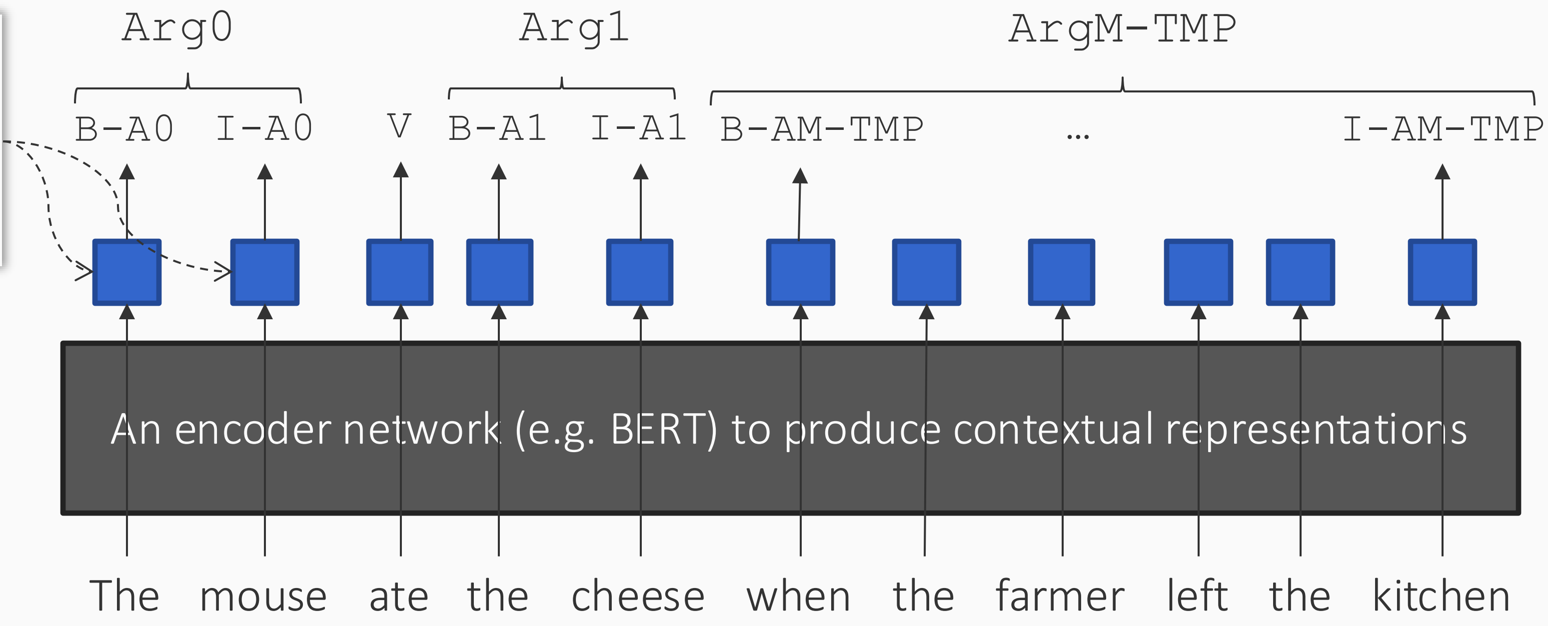
A BIO classifier over the token and verb (a small neural network)



Each predicate treated separately. Here we see the predictions for the verb *ate*.

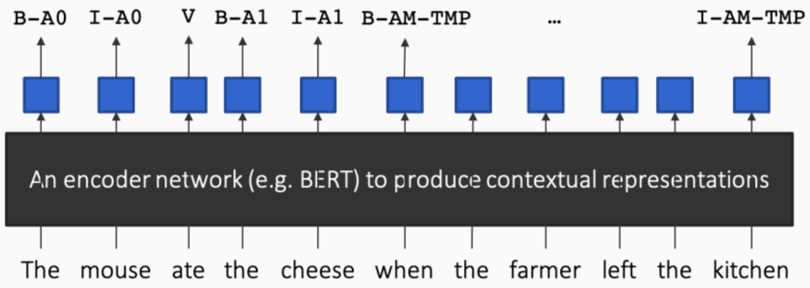
A common design of neural SRL models

A BIO classifier over the token and verb (a small neural network)



A different commonly seen design involves span-based predictions instead of word-level ones.

A remarkable class of models!



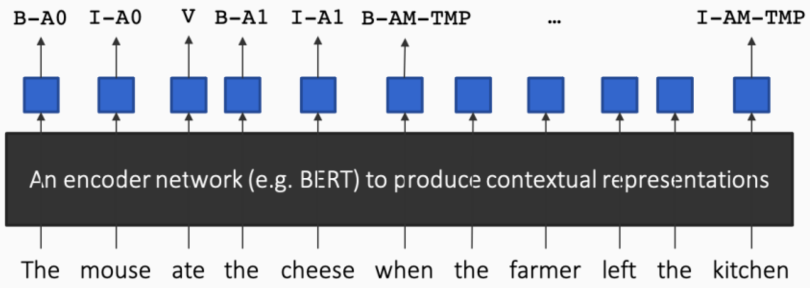
These models are excellent in terms of predictive accuracy

- E.g. with RoBERTa embeddings, on Wall Street Journal data, we can get ~88% F-scores
- And ~80% on out of domain (Brown corpus) sentences

Impressively, they do so with no information about the output space

- No symbolic constraints prohibiting invalid labels
- No structural information about references, etc
- No frame information

A remarkable class of models!



These models are excellent in terms of predictive accuracy

- E.g. with RoBERTa embeddings, on Wall Street Journal data, we can get ~88% F-scores
- And ~80% on out of domain (Brown corpus) sentences

Impressively, they do so with no information about the output space

- No symbolic constraints prohibiting invalid labels
- No structural information about references, etc
- No frame information

Is there room for knowledge in the form of symbolic constraints in neural SRL models?

- If so, how do we introduce it without sacrificing modeling convenience?

Constraints in SRL: Unique Core Roles

For any verb u, and a word i

$$\forall\, u, i \in s$$

Constraints in SRL: Unique Core Roles

For any verb u, and a word i

$$\forall\, u, i \in s$$

$$\boxed{B_X(u,i)}$$

If a model labels the i^{th} word as the beginning of a label X

Constraints in SRL: Unique Core Roles

for any core argument X (i.e. one of A0, A1, A2, A3, A4, A5)

For any verb u, and a word i

$$\forall\, u, i \in s, X \in \mathcal{A}_{core},$$

$$B_X(u, i)$$

If a model labels the i^{th} word as the beginning of a label X

Constraints in SRL: Unique Core Roles

For any verb u, and a word i

for any core argument X (i.e. one of A0, A1, A2, A3, A4, A5)

$$\forall\, u, i \in s, X \in \mathcal{A}_{core},$$

$$B_X(u, i) \rightarrow \boxed{\bigwedge_{\substack{j \in s \\ j \neq i}}}$$

If a model labels the i^{th} word as the beginning of a label X

Then, for any other word j

Constraints in SRL: Unique Core Roles

For any verb u, and a word i

for any core argument X (i.e. one of A0, A1, A2, A3, A4, A5)

$$\forall\, u, i \in s, X \in \mathcal{A}_{core},$$

$$B_X(u,i) \rightarrow \bigwedge_{\substack{j \in s \\ j \neq i}} \neg B_X(u,j)$$

If a model labels the i^{th} word as the beginning of a label X

Then, for any other word j

The model cannot predict that it is the beginning of the same label

Constraints in SRL: Unique Core Roles

For any verb u, and a word i

for any core argument X (i.e. one of A0, A1, A2, A3, A4, A5)

$$\forall\, u, i \in s, X \in \mathcal{A}_{core},$$

$$B_X(u,i) \rightarrow \bigwedge_{\substack{j \in s \\ j \neq i}} \neg B_X(u,j)$$

If a model labels the i^{th} word as the beginning of a label X

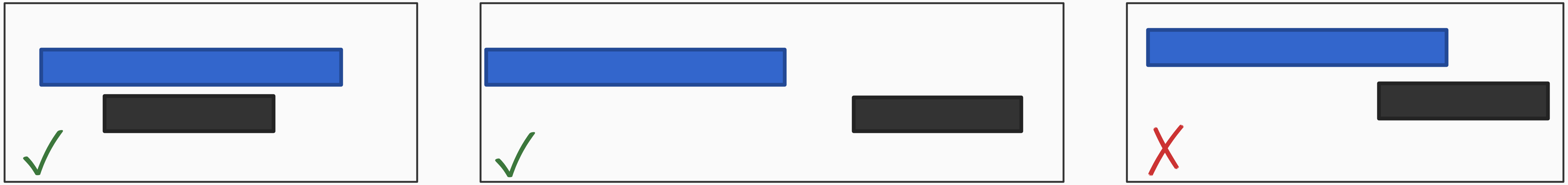
Then, for any other word j

The model cannot predict that it is the beginning of the same label

Other constraints (informally)

The exclusively overlapping role constraint:

– In any sentence, an argument for a predicate can either be contained in, or fully outside, the argument for any predicate



The frame core role constraint

– A verb can have only those core arguments that are defined in PropBank

The modeling questions

How can we incorporate such knowledge into our modeling and/or prediction process?

- The constraints are not dependent on any specific labeled example
- The labeled data presumably already satisfies the constraints

Can such knowledge help?

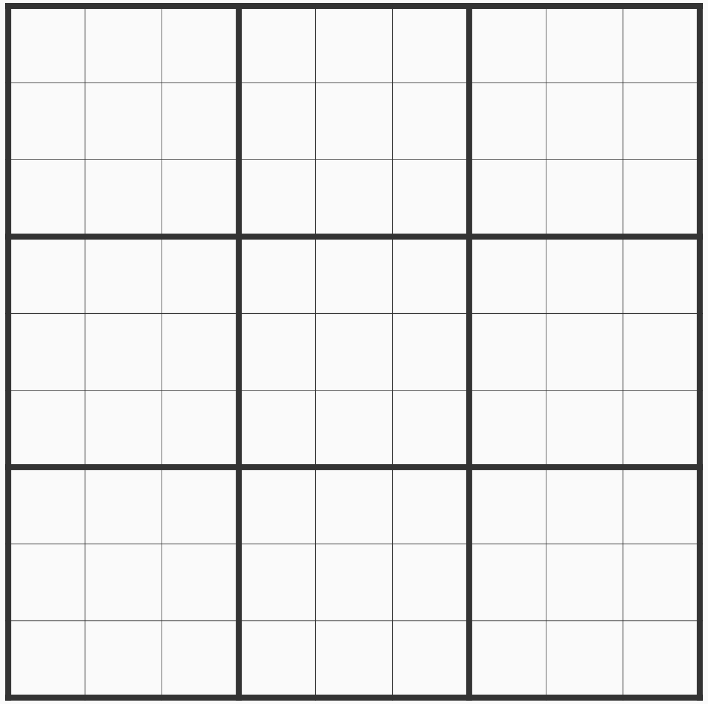
Let’s look at a few examples

1. Recognizing digits and adding them

2. Semantic role labeling

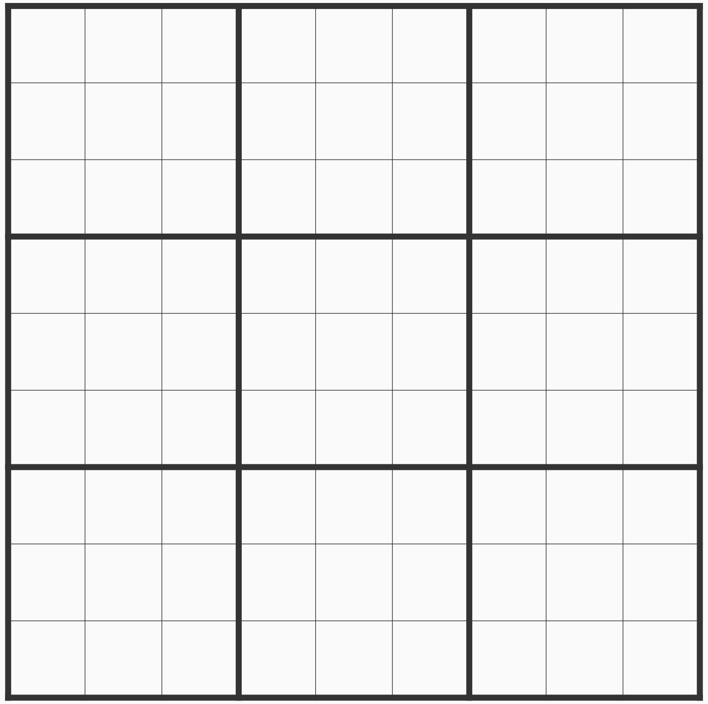
3. Using the rules of a game to play the game

Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

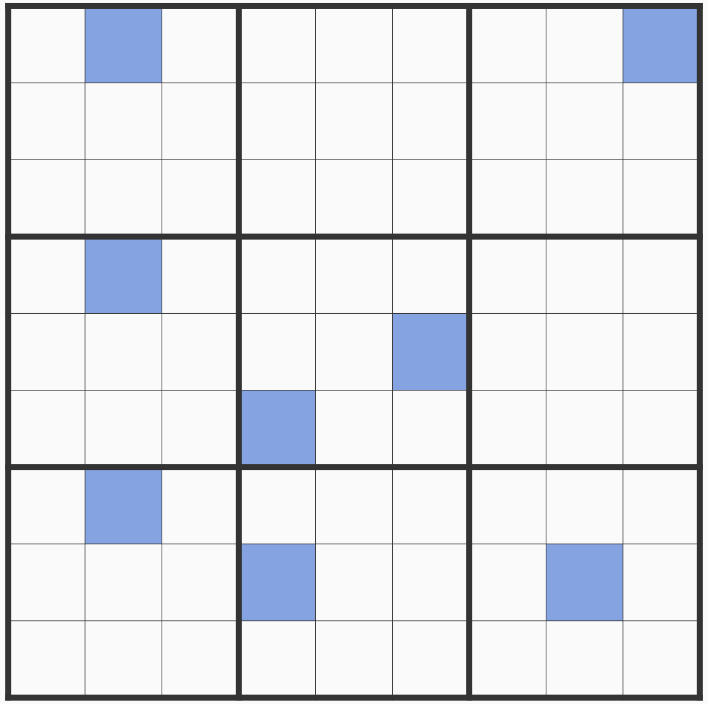
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9

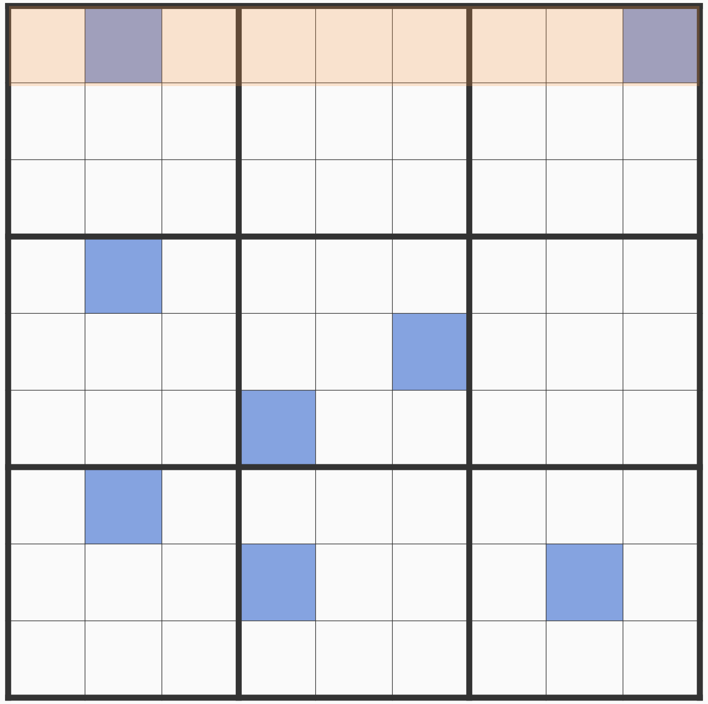
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in

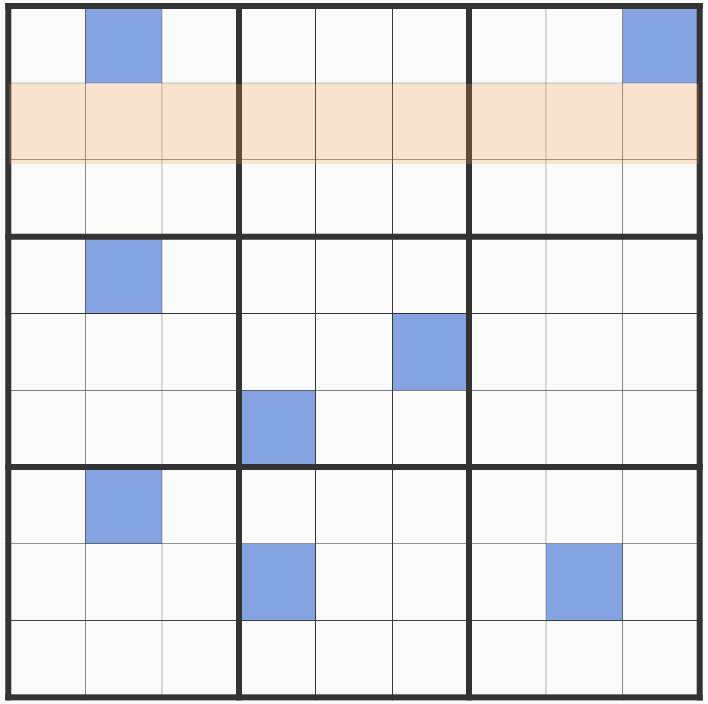
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row

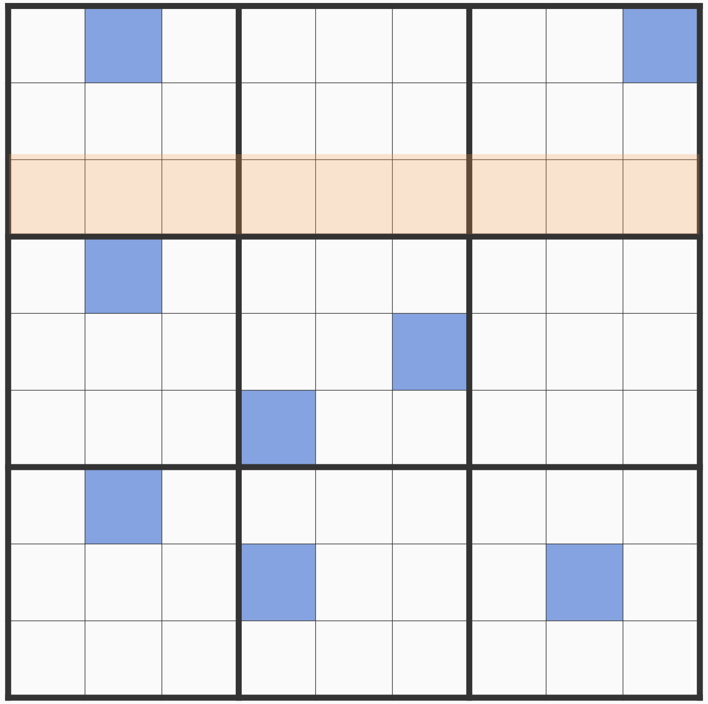
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row

Can we instruct a model to play a game by giving it the rules of the game?



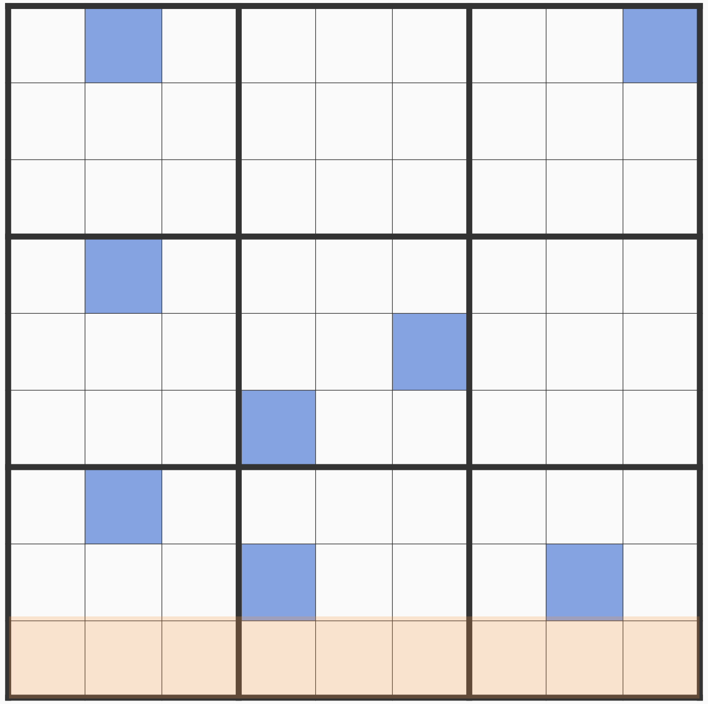
Example: sudoku

1. Each cell can be filled a digit from 1 to 9

2. Some cells are already filled in

3. All digits from 1-9 should show up exactly once in each row

Can we instruct a model to play a game by giving it the rules of the game?

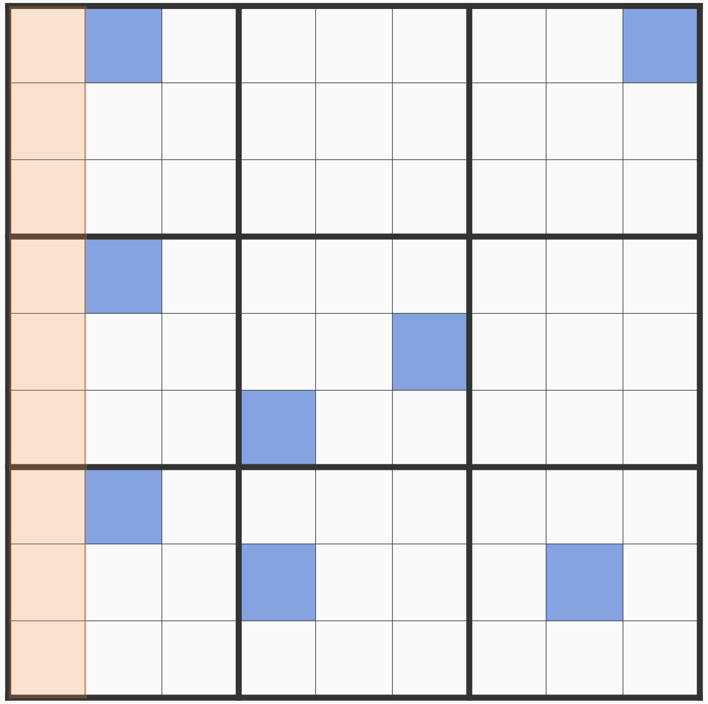


Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row

We have nine rows in all

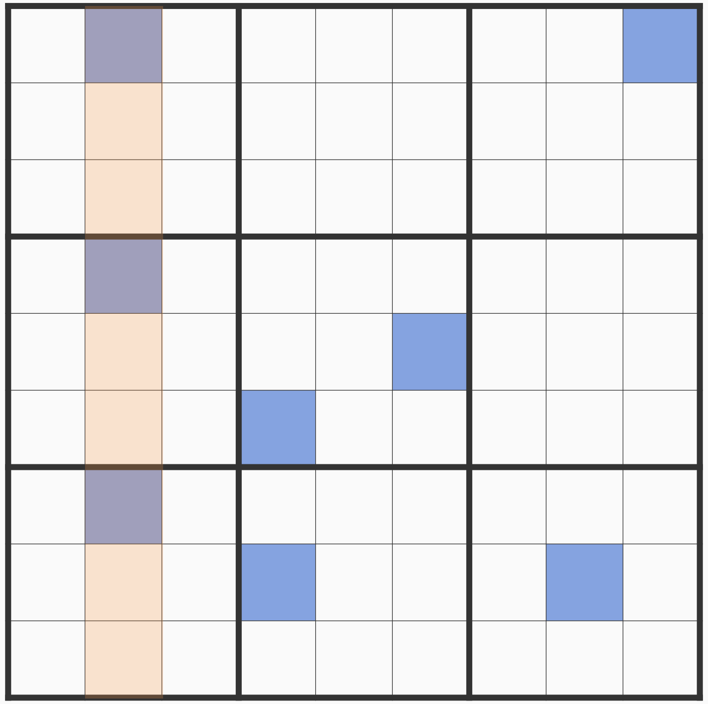
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column

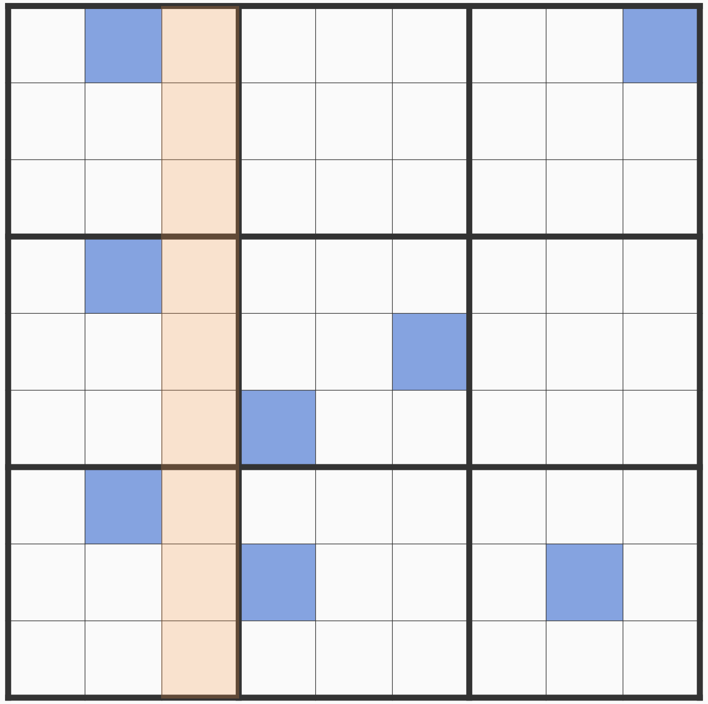
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column

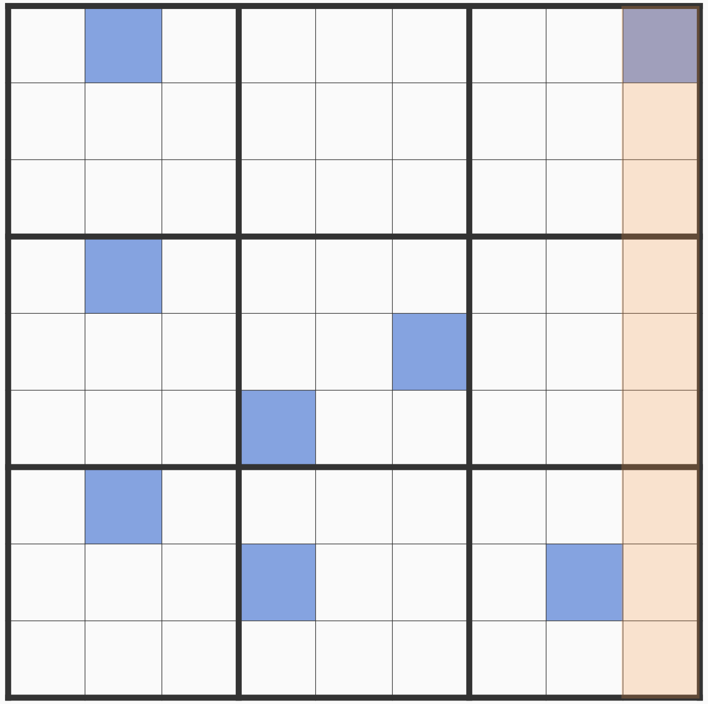
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column

Can we instruct a model to play a game by giving it the rules of the game?

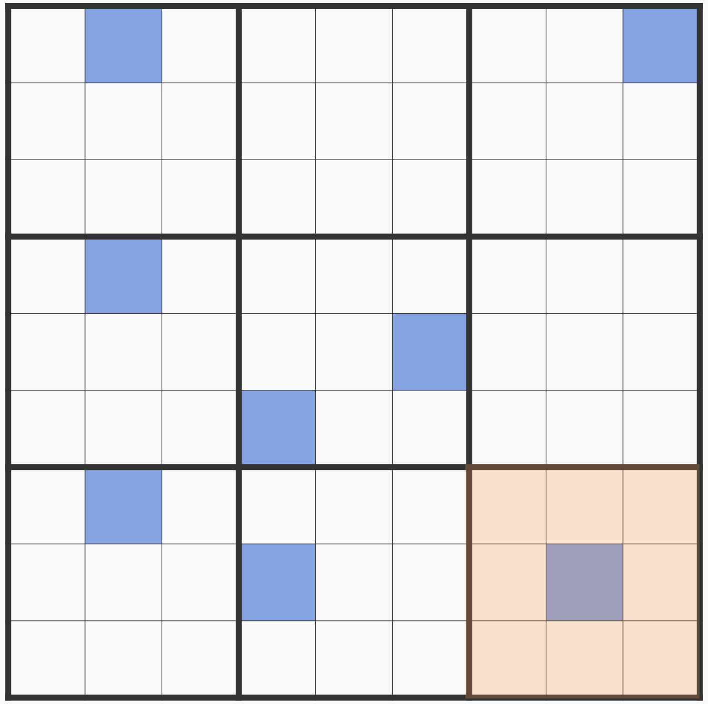


Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column

We have nine columns in all

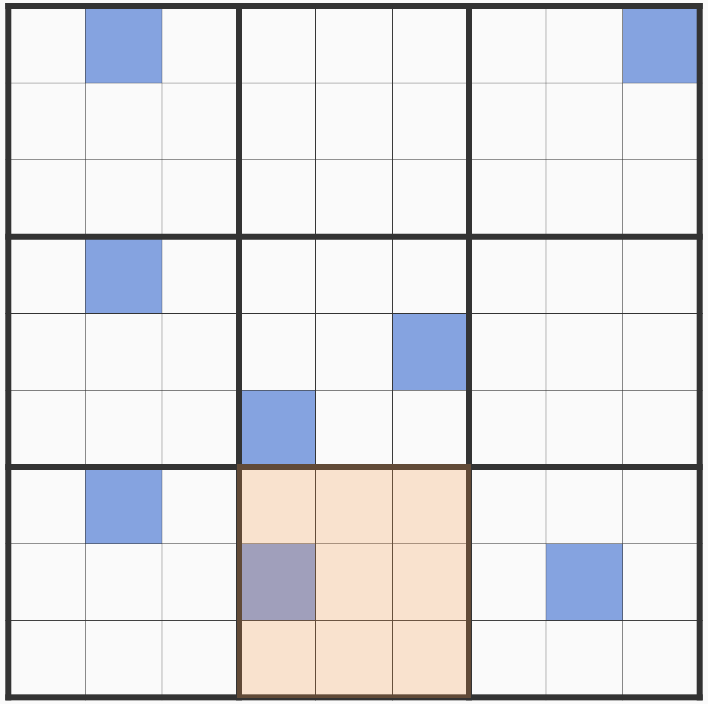
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column
5. All digits from 1-9 should show up exactly once in each 3x3 block

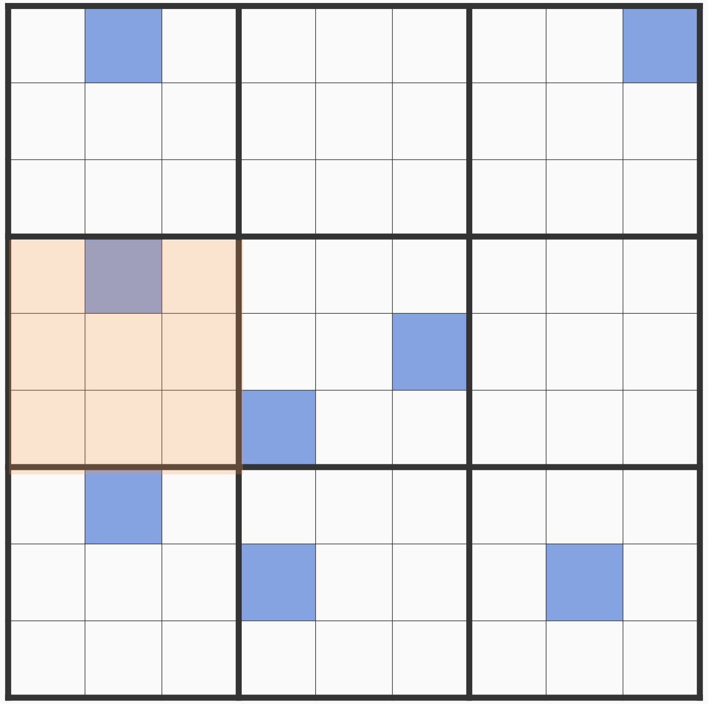
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column
5. All digits from 1-9 should show up exactly once in each 3x3 block

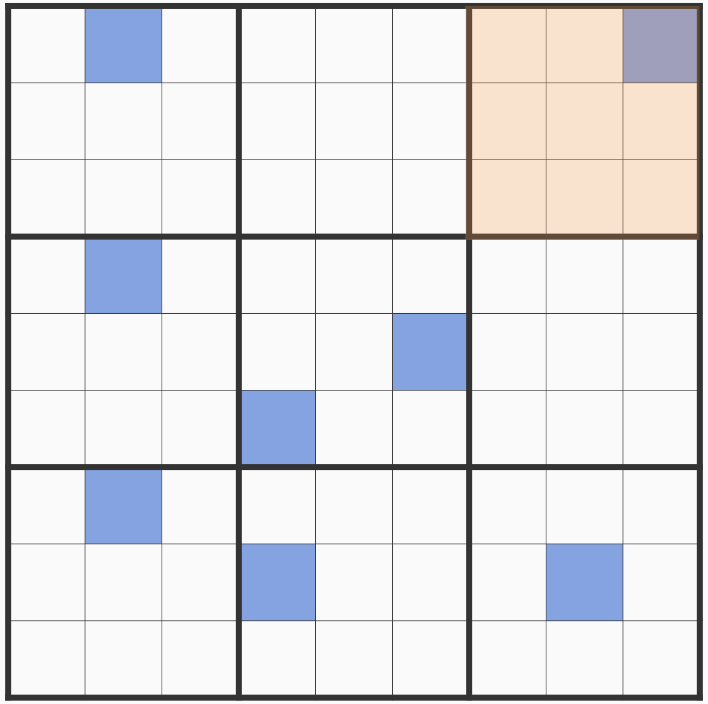
Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column
5. All digits from 1-9 should show up exactly once in each 3x3 block

Can we instruct a model to play a game by giving it the rules of the game?

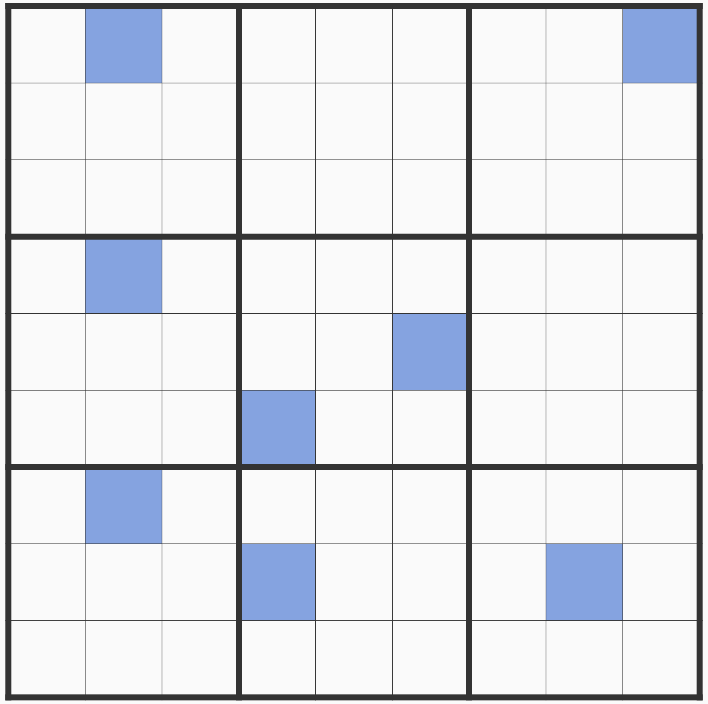


Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column
5. All digits from 1-9 should show up exactly once in each 3x3 block

We have nine blocks in all

Can we instruct a model to play a game by giving it the rules of the game?



Example: sudoku

1. Each cell can be filled a digit from 1 to 9
2. Some cells are already filled in
3. All digits from 1-9 should show up exactly once in each row
4. All digits from 1-9 should show up exactly once in each column
5. All digits from 1-9 should show up exactly once in each 3x3 block

A neural network that can solve sudoku?

Can we train a neural network to solve a sudoku?

Can we do so without any training examples, with only the rules of the game (expressed formally)?

A neural network that can solve sudoku?

Can we train a neural network to solve a sudoku?

Can we do so without any training examples, with only the rules of the game (expressed formally)?

Note: Sudoku is an example of a constraint satisfaction problem. We don't need to train a neural network to solve the problem.

For the purpose of this course, we are asking these questions to demonstrate what *can* be done, not necessarily what *should* be done

Let’s look at a few examples

1. Recognizing digits and adding them

2. Semantic role labeling

3. Using the rules of a game to play the game

4. Other examples

Other motivating examples

Can we construct a knowledge base using facts learned from the internet and also knowledge about the world such as "The father of a father is a grandfather"?

Can we write a program that operates over multiple independent large language model predictions to produce an output that is consistent with some pre-defined rules?

Can we train a neural network with only a limited amount of training data for a task, given that we have some knowledge about the task?