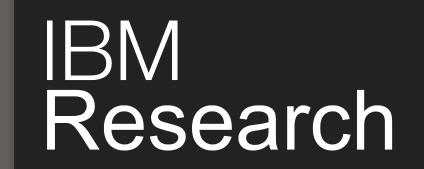


Ariadne



Types for Machine Learning Allison Allain, Avraham Shinnar, Julian Dolby, and Jenna Reinen

SUPPORTIVE TOOLING FOR ML

Traditional Python

No type information for x Extensive type comments Tedious **Error Prone**

Python with Ariadne

Type information exposed

Type comments unnecessary Effortless

Automatically Maintained

MNIST data input is a 1-D vector of 784 features (28*28 pixels) # Reshape to match picture format [Height x Width x Channel] # Tensor input become 4-D: [Batch Size, Height, Width, Channel] z = tf.reshape(x, shape=[-1, 28, 28, 1])

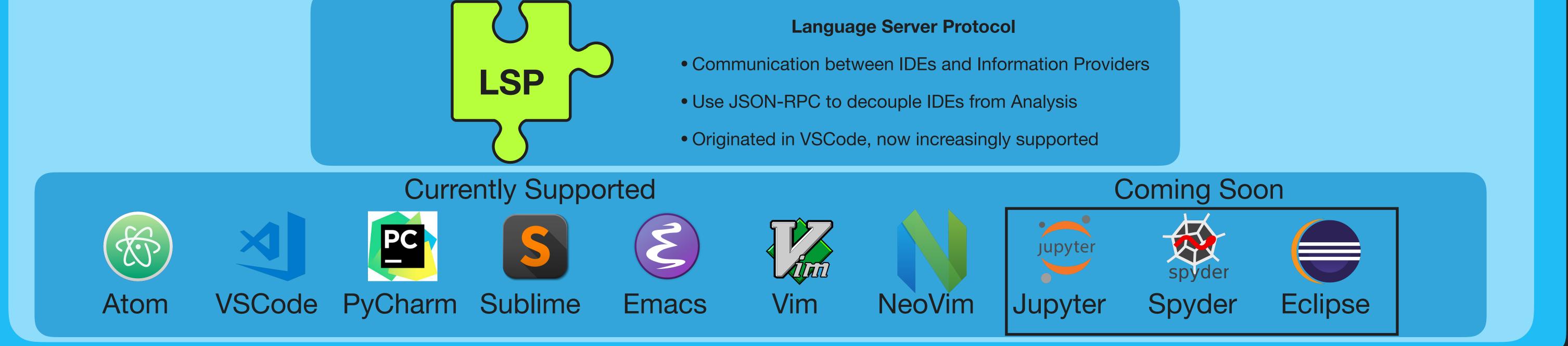
Convolution Layer with 32 filters and a kernel size of 5 conv1 = tf.layers.conv2d(z, 32, 5, activation=tf.nn.relu) # Max Pooling (down-sampling) with strides of 2 and kernel size of 2 conv1 = tf.layers.max_pooling2d(conv1, 2, 2)

41	# MNIST data input is a 1–D vector of 784 features (28*28 pixels)
42 43	<pre>x type: pixel[n][28 * 28] h picture format [Height x Width x Channel] come 4-D: [Batch Size, Height, Width, Channel]</pre>
44	z = tf.reshape(x, shape=[-1, 28, 28, 1])
45 46	z type: pixel[?][28][28][1] $ imes$, with 32 filters and a kernel size of 5
47	<pre>conv1 = tf.layers.conv2d(z, 32, 5, activation=tf.nn.relu)</pre>
48	<pre># Max Pooling (down-sampling) with strides of 2 and kernel size of 2</pre>
49	<pre>conv1 = tf.layers.max_pooling2d(conv1, 2, 2)</pre>
50	

IMPLEMENTATION



- full-featured, state-of-the-art analysis framework
- used for JavaScript in tools such as AppScan
- extend to analyze Python, using flexible CAst front end



AVAILABLE FEATURES

 • represent the flow of data • track tensors flowing through the program • in this case, globally through object 'x_dict" 	 buggy_convolutional_network.py(37, 13): x_dict['images'] buggy_convolutional_network.py(33, 14): x_dict buggy_convolutional_network.py(78, 28): features buggy_convolutional_network.py(73, 14): features buggy_convolutional_network.py(117, 7): {'images': mnist.train.images} buggy_convolutional_network.py(117, 18): mnist buggy_convolutional_network.py(117, 18): mnist.train.images 	 Uses WALA slicer Whole program Flow & context sensitive 		
 expose the shape of assigned tensor shown describe data throughout the program 	<pre>z: pixel[?][28][28][1] z = tf.reshape(x, shape=[-1, 28, 28, 1]) conv1: pixel[?][28][28][1] # Convolution Layer with 32 filters and a kernel size of 5 conv1 = tf.layers.conv2d(z, 32, 5, activation=tf.nn.relu)</pre>	Uses WALA dataflow Propagatos topsor shapes		
 show tensor variables display when users hover over code provide detail only when requested 	<pre>28][28][1] reshape(x, shape=[z type: pixel[?][28][28][1] ution Layer with tf.layers.conv2d(2, 32, 5, activation=tf.nn.re)</pre>	 Propagates tensor shapes Usual fixpoint dataflow Models tensor operations 		
• integrate with IDE error reporting	bad_conv1 = tf.layers.conv2d(k, 32, 5, activation=tf.nn.relu) tf.reshape(x, [-1, 28, 28, 1]) gy_convolutional_network.py ~/git/ML/com.ibm.wala.cast.python.test/data 4 wriadne] Cannot reshape pixel[n][28 * 28] to pixel[?][11][28][1] (38, 28) wriadne] Bad type to convolve pixel[n][28 * 28], needs 4 dimensions (possible fix: tf.reshape(x, [-1, 28, 28 (55, 38))	 Tensor models for misuse Heuristics to suggest fixes 		
https://wala.github.io/ariadne				