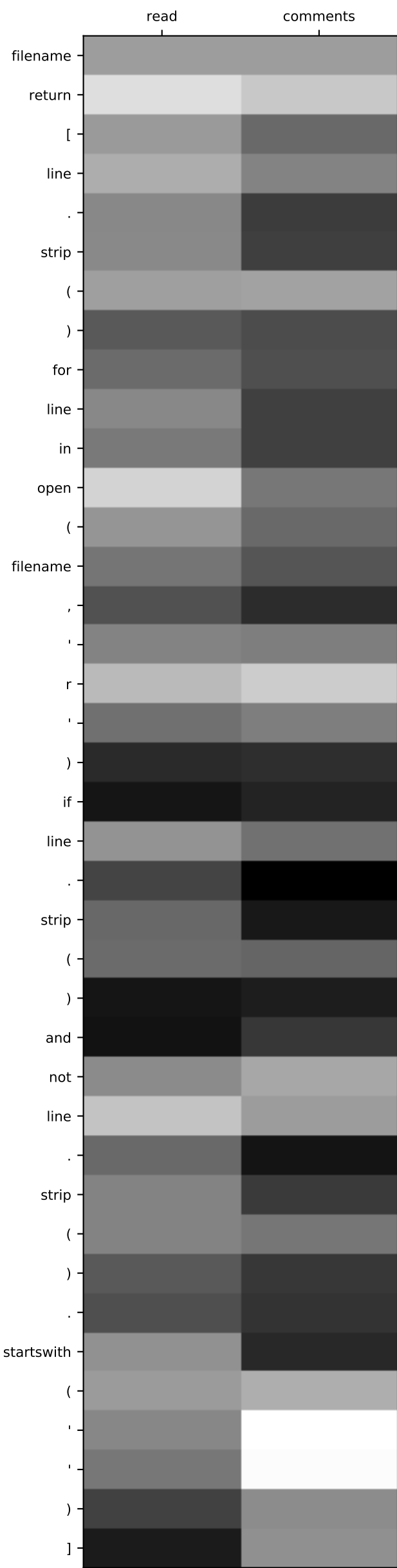
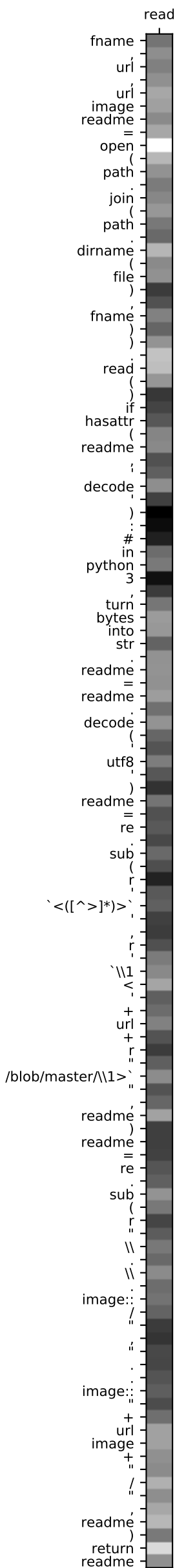


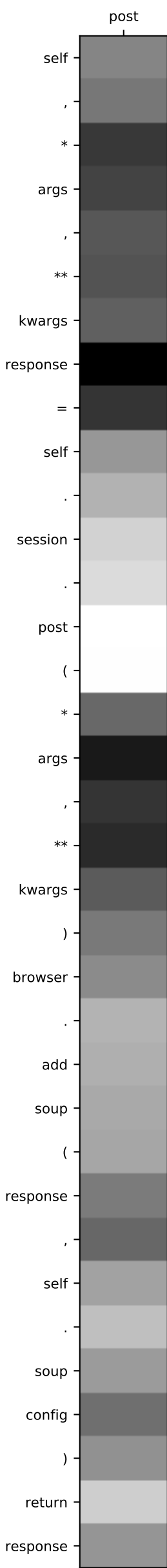
Ground truth:requirements from file



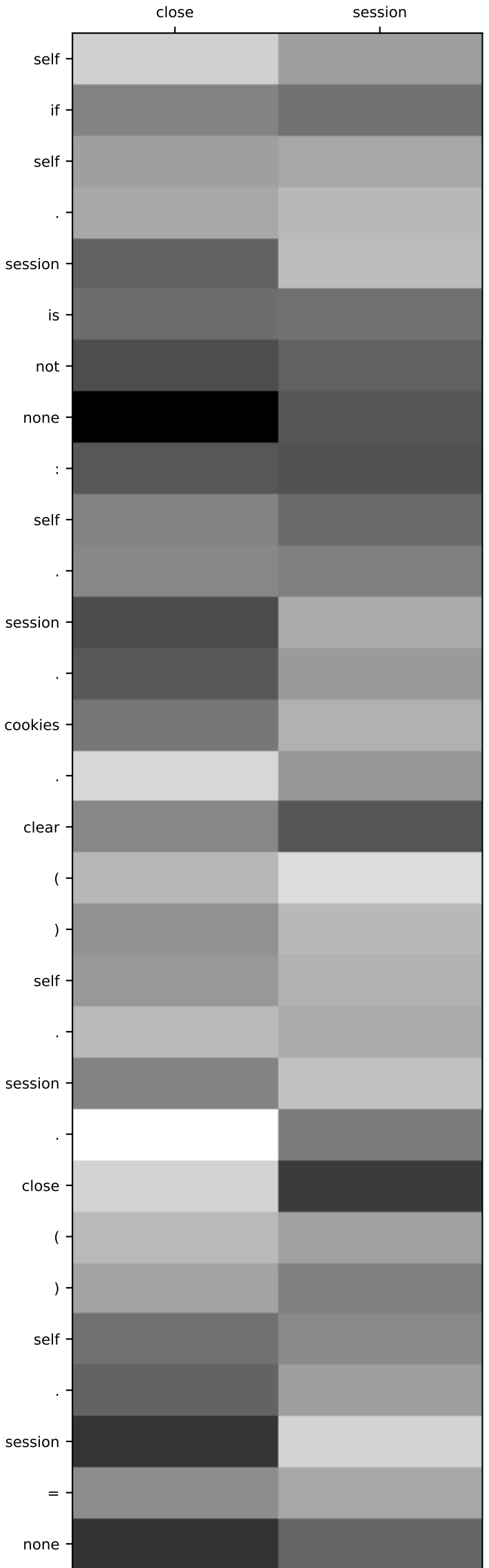
Ground truth:read



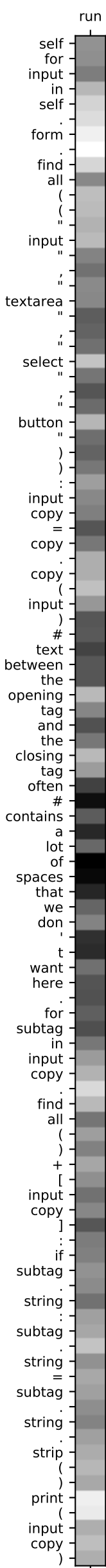
Ground truth:post



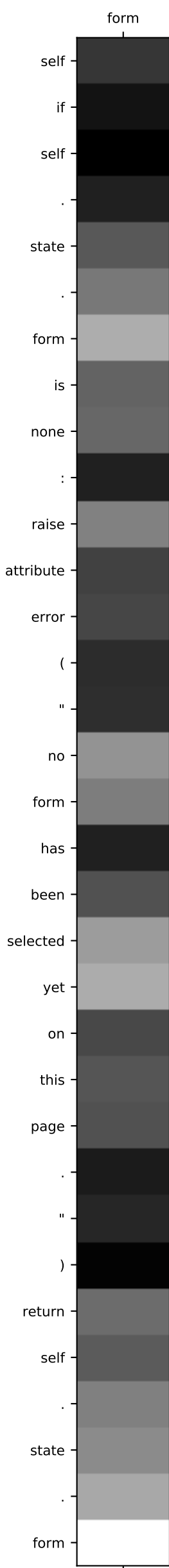
Ground truth:close



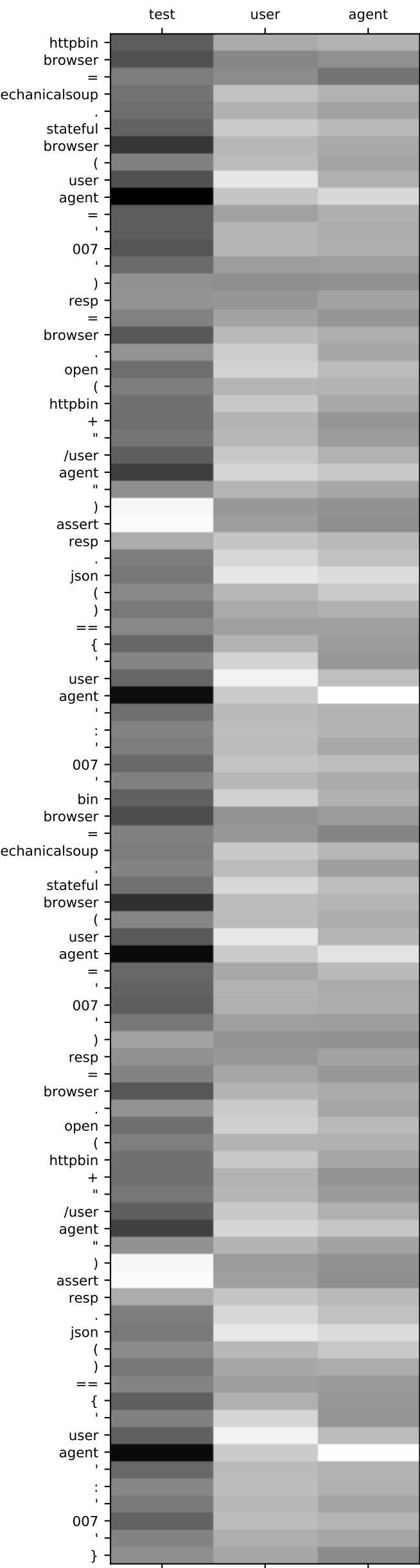
Ground truth:print summary



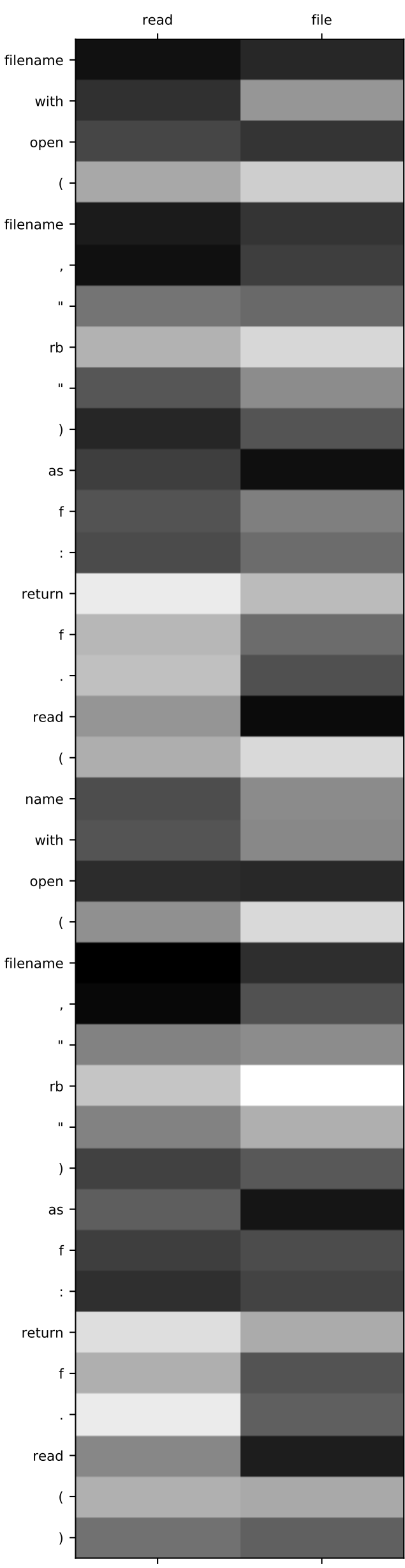
Ground truth:form



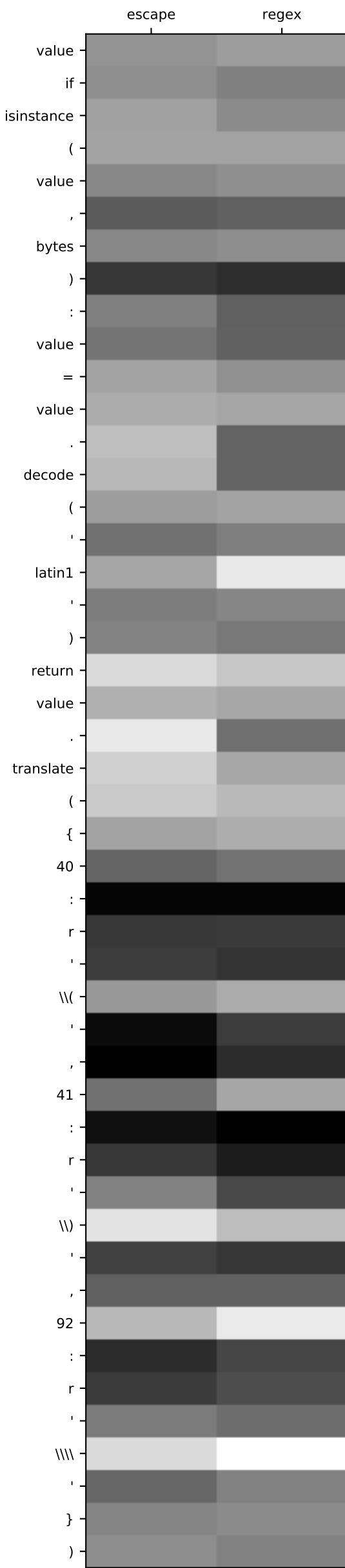
Ground truth:test user agent



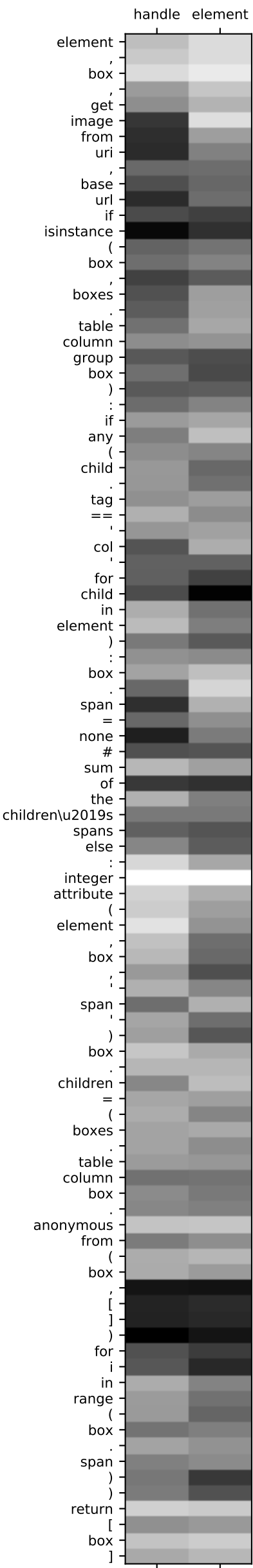
Ground truth:file get contents



Ground truth:pdf escape

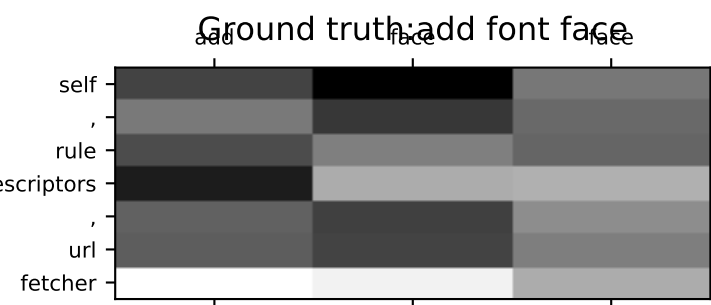


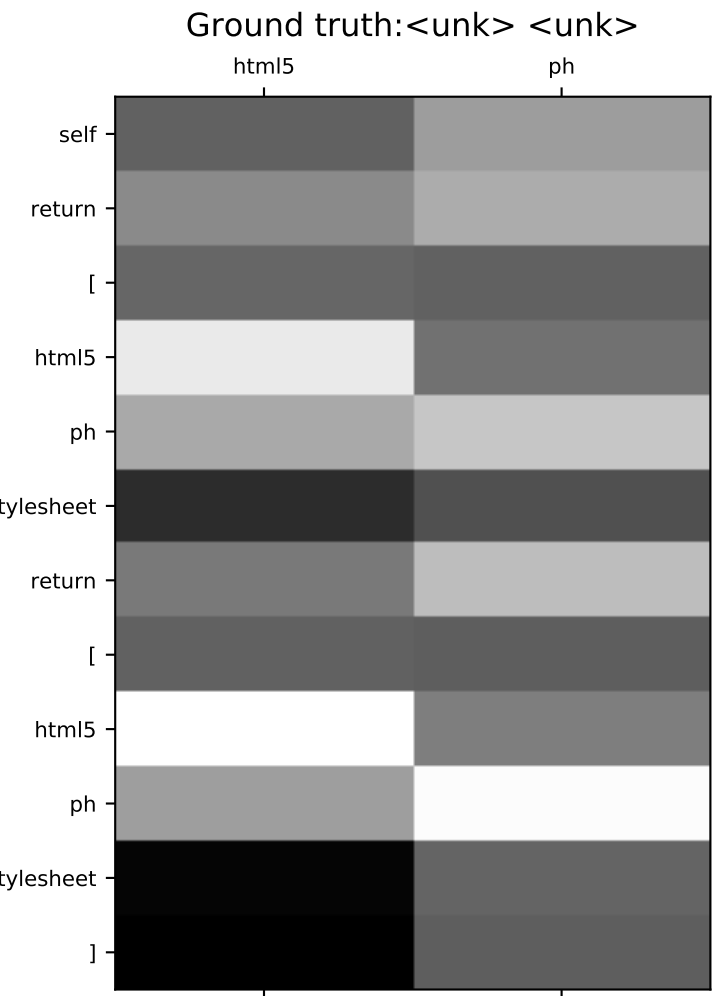
Ground truth:handle <unk>



Ground truth:from exception

	from	exception
cls		
,		
exception		
name		
=		
type		
(
exception		
)		
.		
name		
value		
=		
str		
(
exception		
)		
return		
cls		
(
,		
%s:		
%s		
,		
%		
(
name		
,		
value		
)		
if		
value		
else		
name		
,		
exception		
name		
=		
type		
(
exception		
)		
.		
name		
value		
=		
str		
(
exception		
)		
return		
cls		
(
,		
%s:		
%s		
,		
%		
(
name		
,		
value		
)		
if		
value		
else		
name		
)		

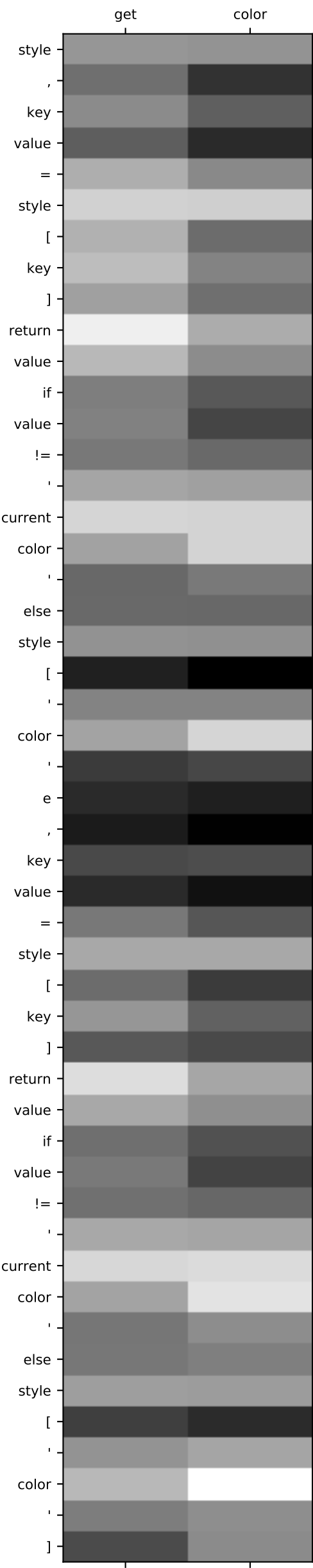




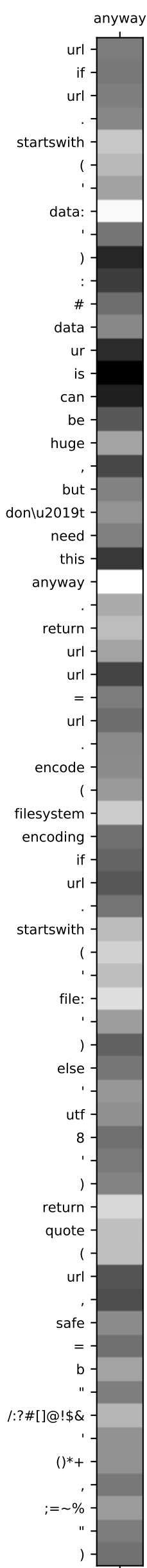
Ground truth:write pdf



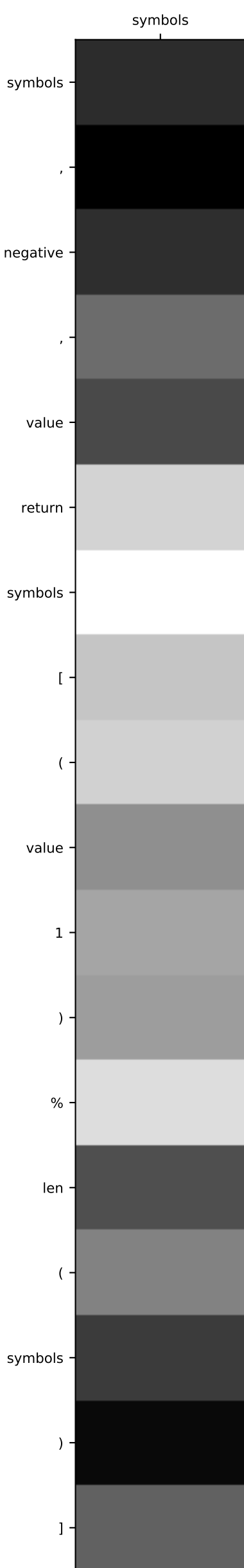
Ground truth:get color



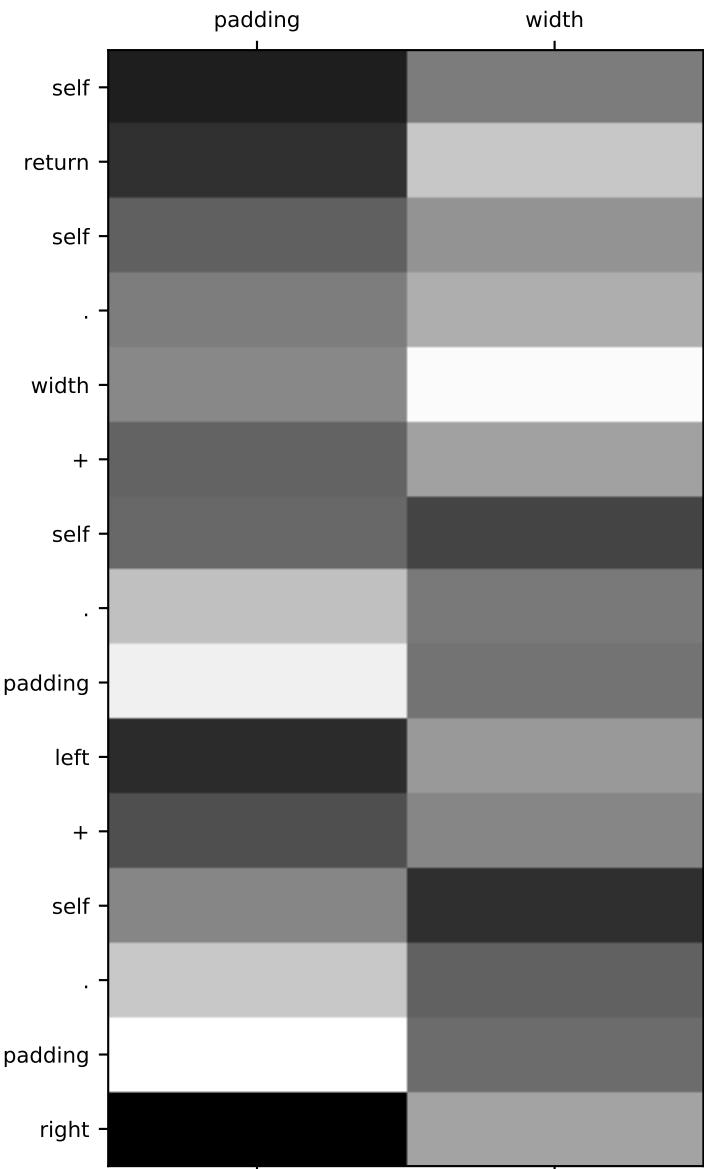
Ground truth:<unk> to uri



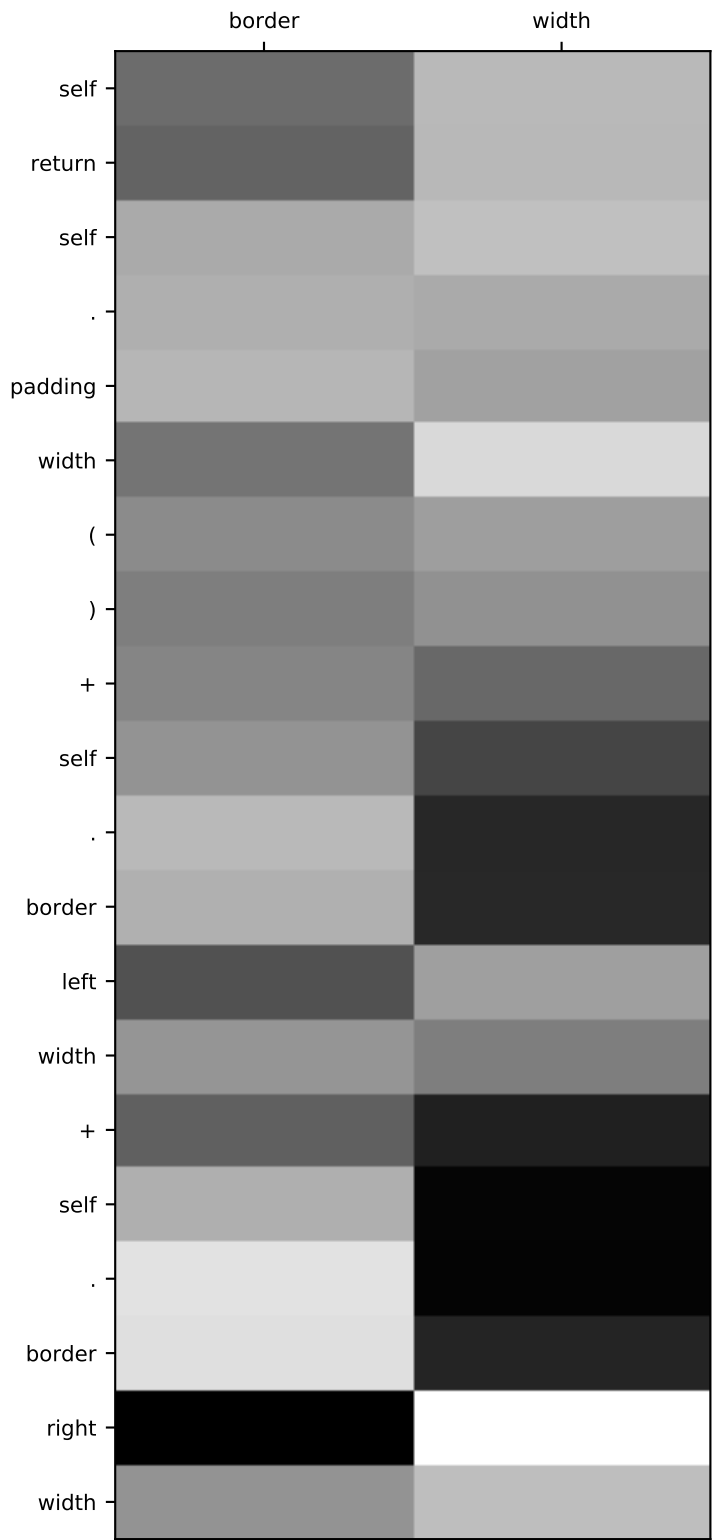
Ground truth:<unk>



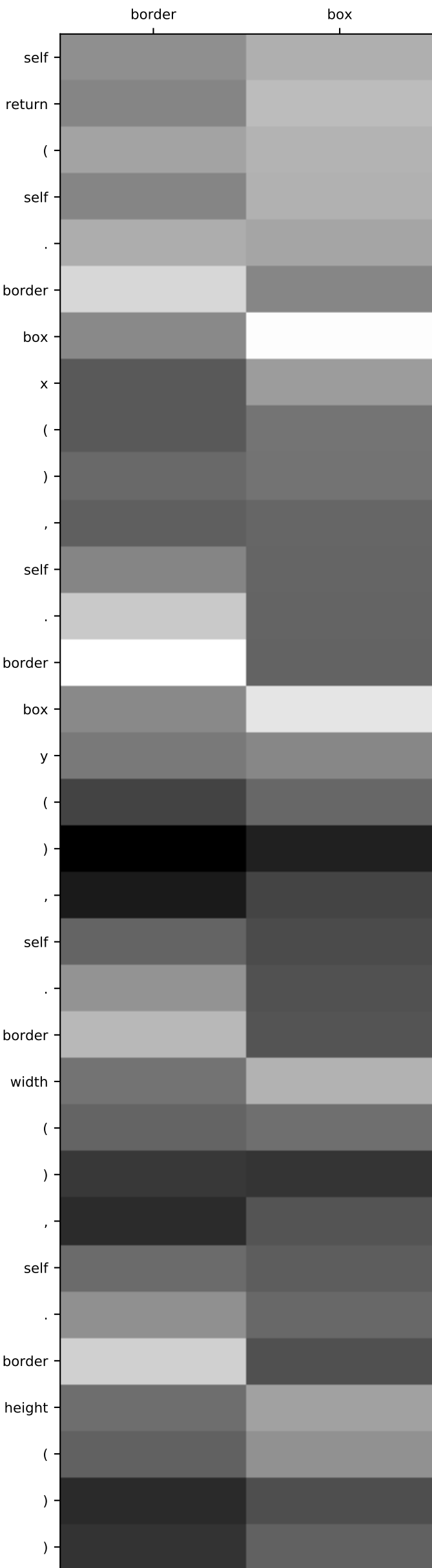
Ground truth:padding width



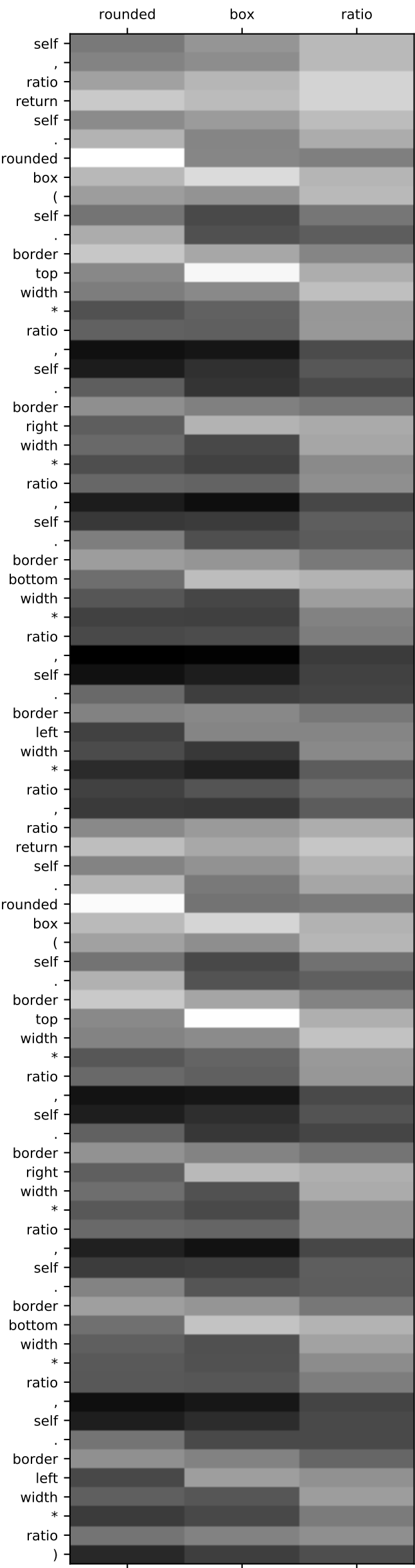
Ground truth:border width



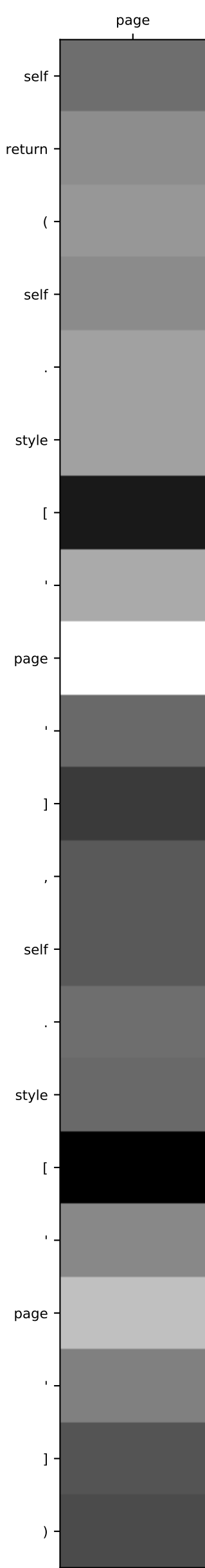
Ground truth:hit area



Ground truth:<unk> box ratio



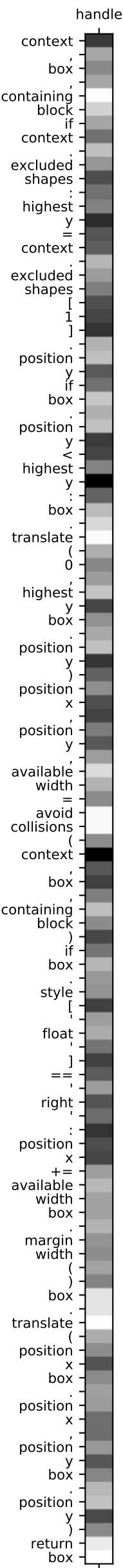
Ground truth:page values



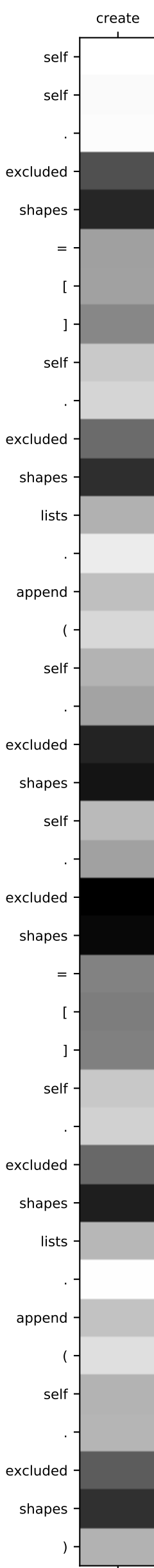
Ground truth:table cell min content width

	cell	min	content	width
context				
box'				
outer'				
children				
widths				
=				
[
min				
content				
width				
(
context				
child				
outer'				
=				
true				
)				
for				
child				
in				
box				
.				
children				
if				
not				
child				
.				
is				
absolutely				
positioned				
{				
}				
children				
min				
width				
=				
margin				
width				
(
box				
'				
max				
(
children				
widths				
)				
if				
children				
widths				
else				
0				
)				
width				
=				
box				
.				
style				
{				
width				
,				
}				
if				
width				
!=				
,				
auto				
,				
and				
width				
.				
unit				
==				
,				
px				
,				
:				
cell				
min				
width				
=				
adjust				
(
box				
outer'				
width'				
.				
value				
)				
else				
:				
cell				
min				
width				
=				
0				
return				
max				
(
children				
min				
width				
.				
cell				
min				
width				
)				

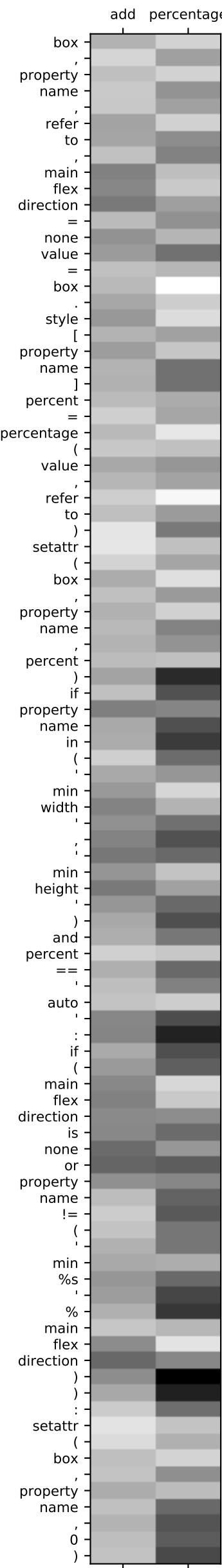
Ground truth:find float position



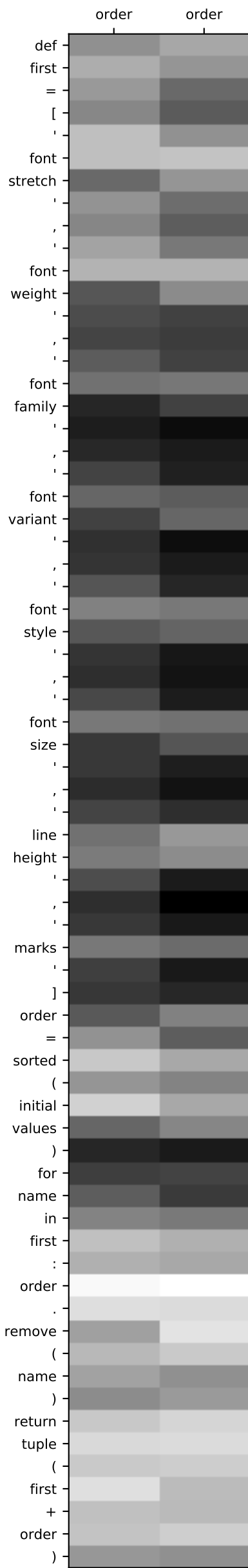
Ground truth:create block formatting context



Ground truth:resolve one percentage



Ground truth:<unk> order



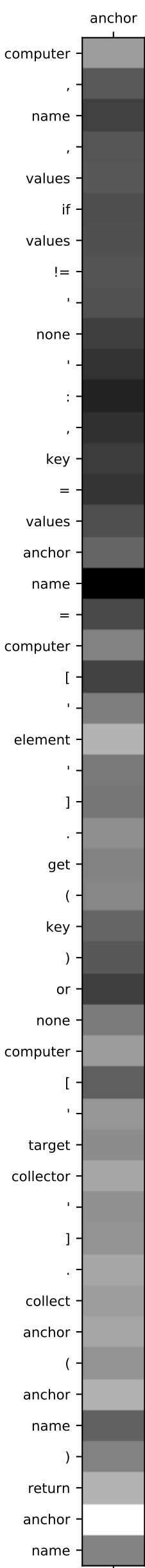
Ground truth:length tuple

	length	tuple
computer		
,		
name		
,		
values		
return		
tuple		
(
length		
(
computer		
,		
name		
,		
value		
,		
pixels		
only		
=		
true		
)		
for		
value		
in		
values		
)		

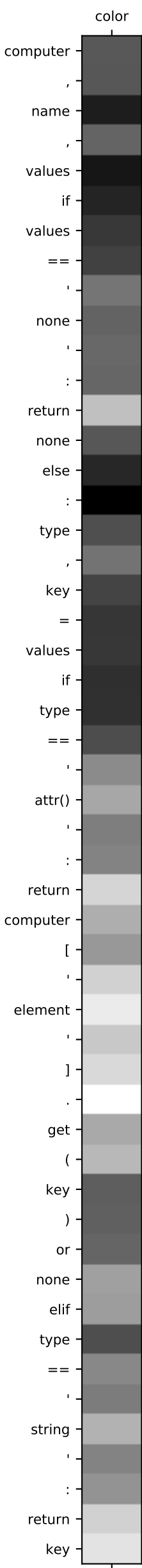
Ground truth:font weight



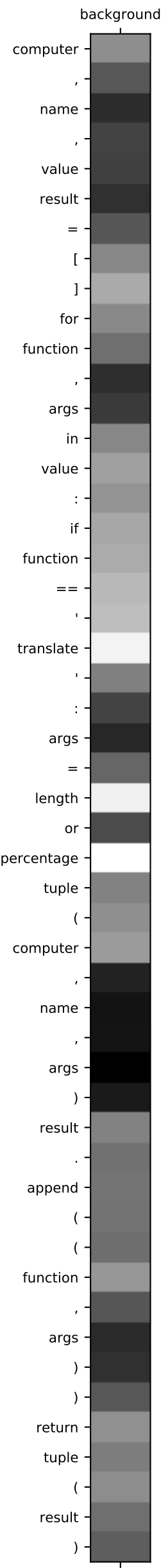
Ground truth:anchor



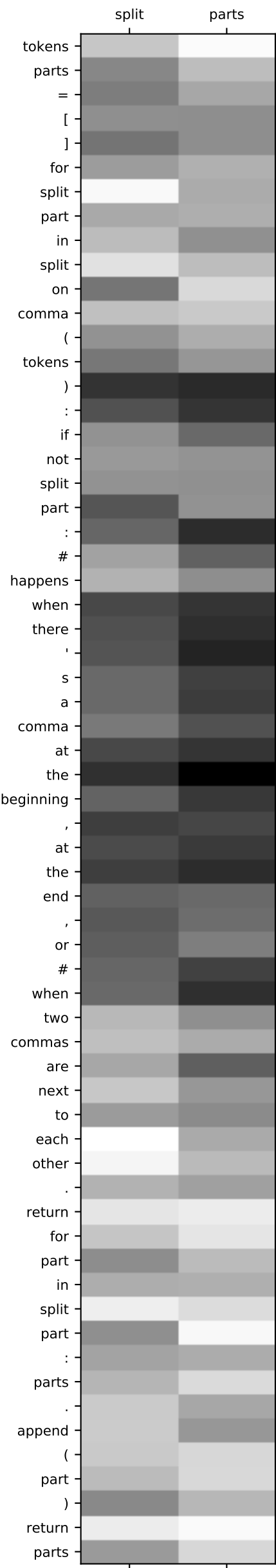
Ground truth:lang



Ground truth:transform

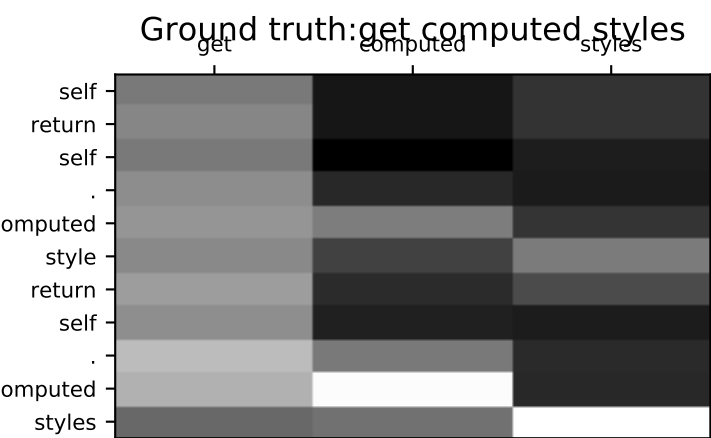


Ground truth:split on optional comma

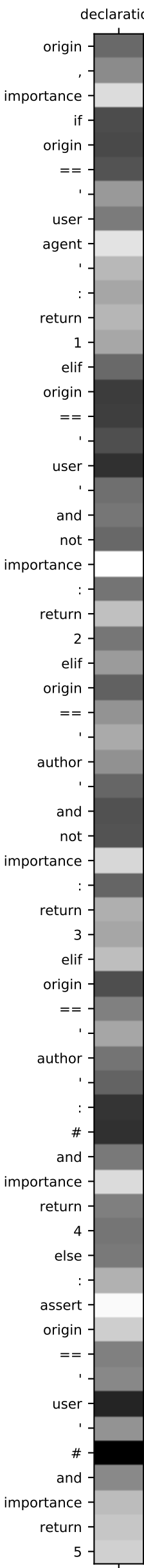


Ground truth:get angle

	get	angle
token		
if		
token		
.		
type		
==		
'		
dimension		
'		
:		
factor		
=		
angle		
to		
radians		
.		
get		
(
token		
.		
unit		
)		
if		
factor		
is		
not		
none		
:		
return		
token		
.		
value		
*		
factor		



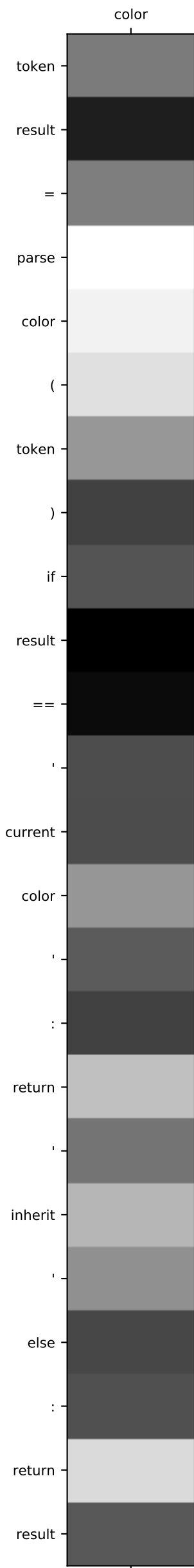
Ground truth:declaration precedence



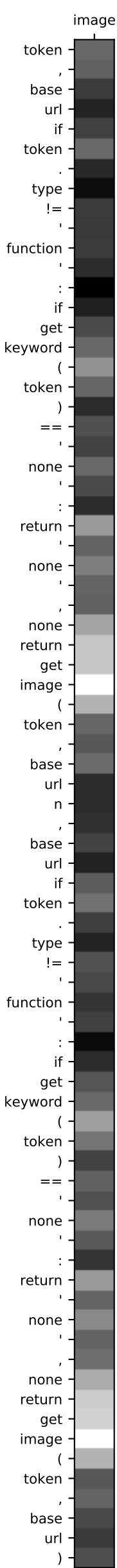
Ground truth:store target



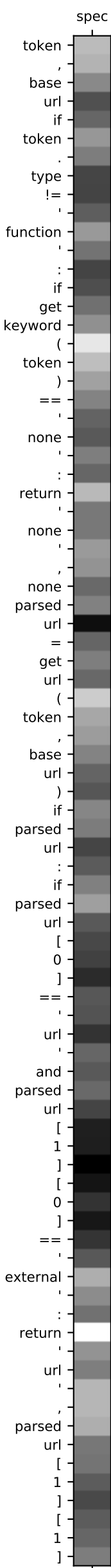
Ground truth:color



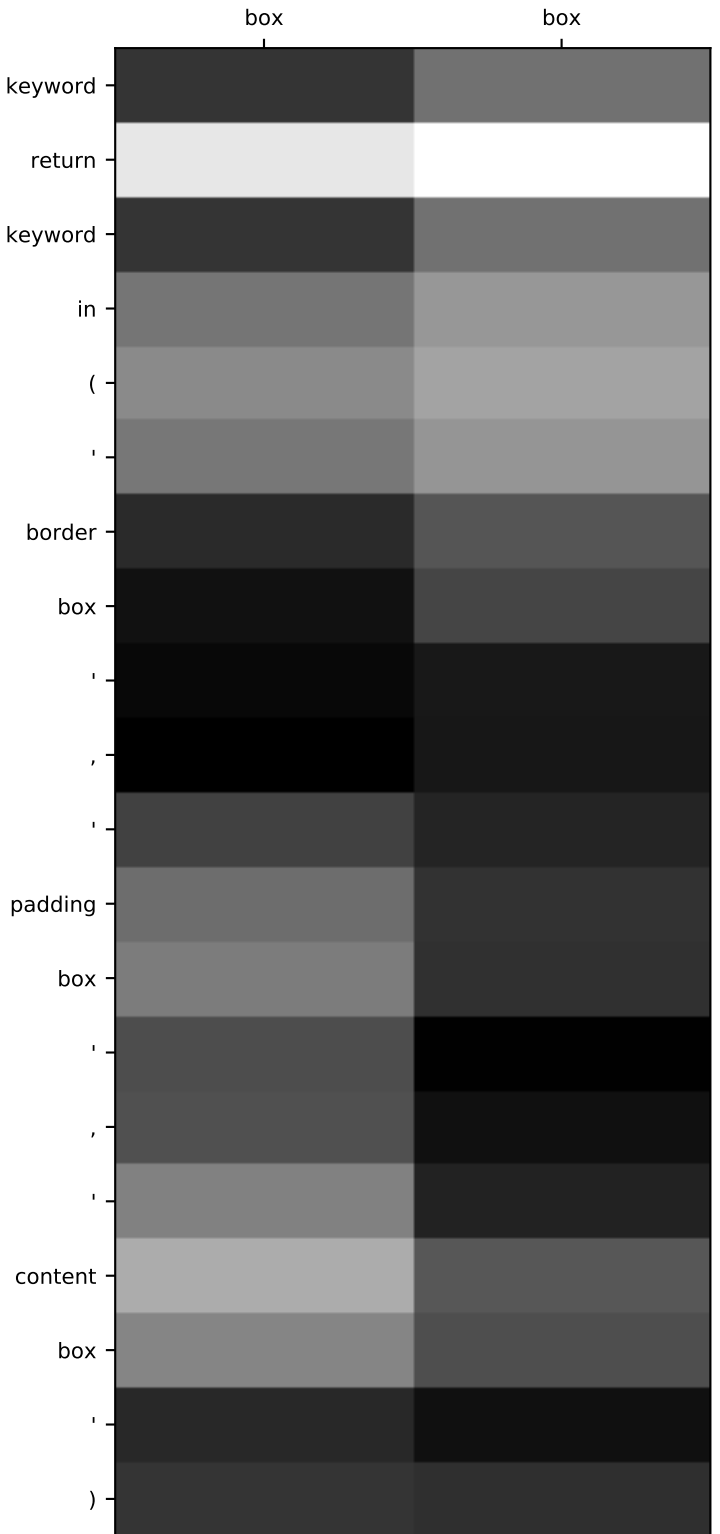
Ground truth:background image



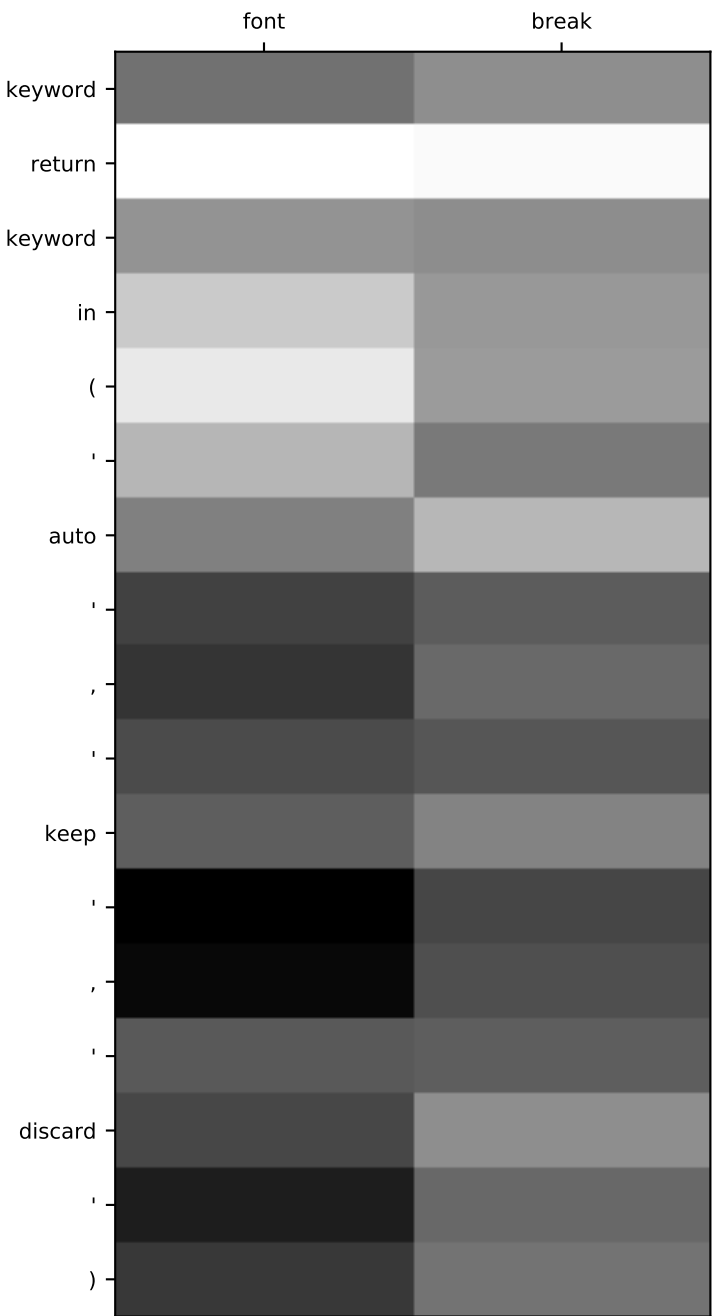
Ground truth:list style image



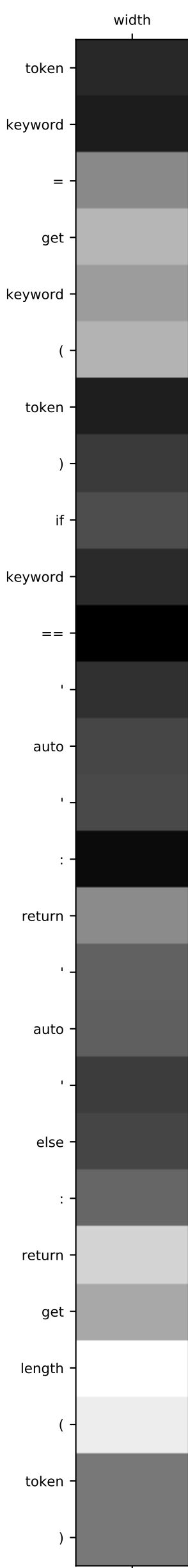
Ground truth:box



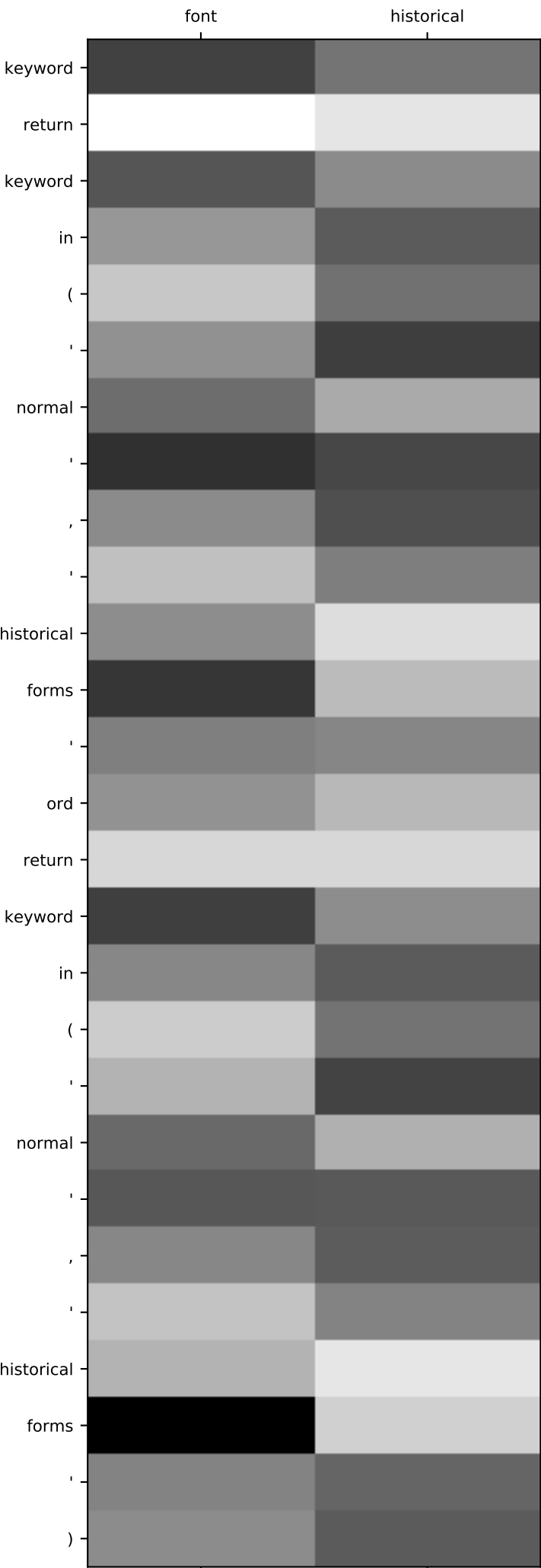
Ground truth:margin break



Ground truth:<unk>



Ground truth:font variant <unk>

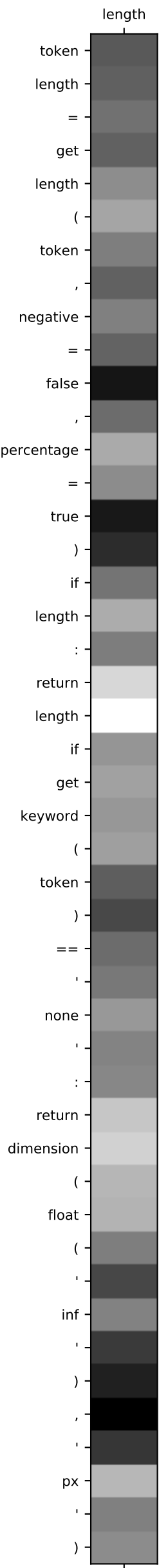


Ground truth:font weight

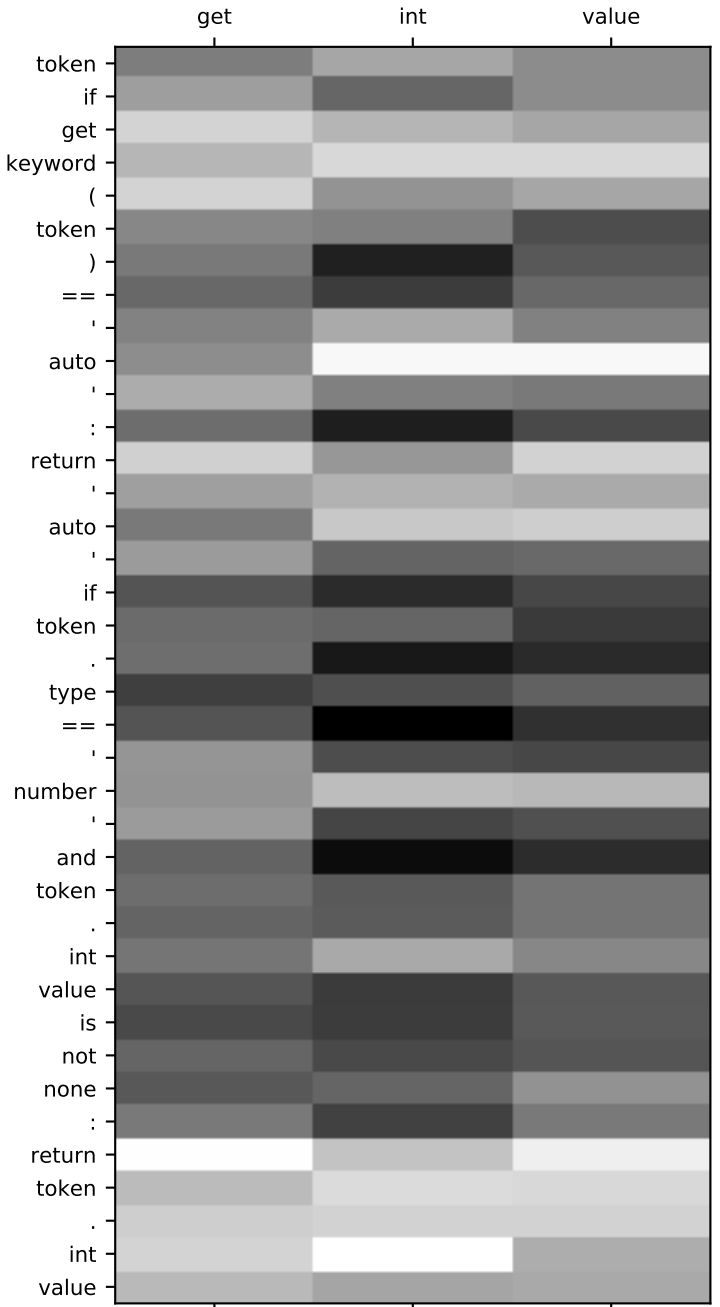
normal

```
token
keyword
=
get
keyword
(
token
)
if
keyword
in
(
'
normal
'
,
'
bold
'
,
'
bolder
'
,
'
lighter
'
)
:
return
keyword
if
token
.
type
==
'
number
'
and
token
.
int
value
is
not
none
:
if
token
.
int
value
in
(
100
,
200
,
300
,
400
,
500
,
600
,
700
,
800
,
900
)
:
return
token
.
int
value
```

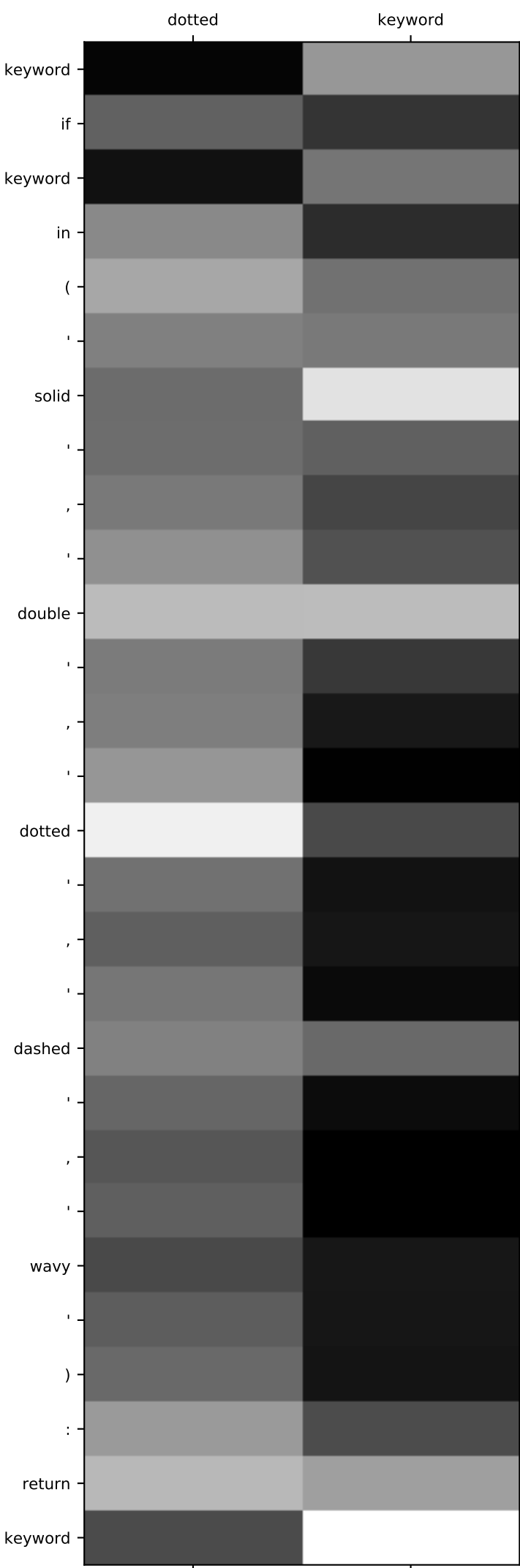
Ground truth:max width height



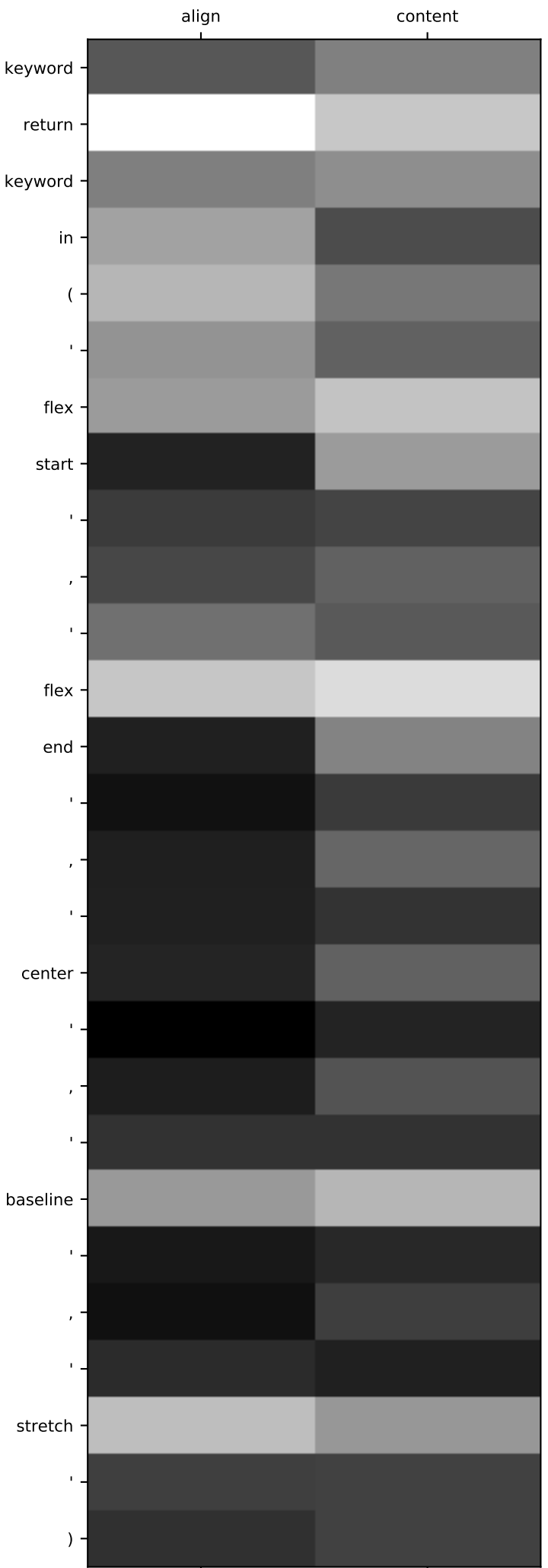
Ground truth:z index



Ground truth:text <unk> style



Ground truth:align items



Ground truth:run

