

Localización 2D de yemas de vid utilizando técnicas de deep learning

Wenceslao Villegas Marset^{a,*}, Diego Sebastián Pérez^{a,b}, Carlos Ariel Díaz^a,
Facundo Bromberg^{a,b}

^aUniversidad Tecnológica Nacional, Facultad Regional Mendoza, Grupo de Inteligencia Artificial DHARMA, Dpto. de Sistemas de la Información. Rodríguez 273, CP 5500, Mendoza, Argentina.

^bConsejo Nacional de Investigaciones Científicas y Técnicas (CONICET) .

Abstract

TODO: Abstract

Keywords: Computer vision, Fully Convolutional Network, Grapevine bud detection, Precision viticulture

1. Introduction

In this work we propose a solution for the autonomous detection of grapevine buds within 2D images of vineyards captured in natural field conditions. Our proposed approach is based on *Fully Convolutional Networks (FCN)* (Long et al., 2015; Shelhamer et al., 2017), a kind of deep learning model specific for computer vision applications. Our solution adds in the historical quest for more and better quality information about different vineyard processes that impact on the productivity of grapevines and quality of their grapes.

For centuries agronomists have been producing models of the most relevant plant processes (i.e. fruit quality and yield, soil profiling, vine health), and vineyard managers have been recollecting a diverse corpus of information for feeding these models. Better and more efficient measuring procedures resulted in more information with its corresponding impact on the quality of models' outcomes, while inspiring researchers to push the boundaries for producing more

*Corresponding author

Email addresses: diego.villegas@alumnos.frm.utn.edu.ar (Wenceslao Villegas Marset), sebastian.perez@frm.utn.edu.ar (Diego Sebastián Pérez), carlos.diaz@frm.utn.edu.ar (Carlos Ariel Díaz), fbromberg@frm.utn.edu.ar (Facundo Bromberg)

sophisticated models. Such information consists of a large set of variables for assessing different aspects of the parts of the plant involved in these processes: trunks, leaves, berries, buds, shoots, flowers, bunches, canes. Nowadays technology is pushing once again the possibilities in the quality and throughput of these measurements, with digital and autonomous measurement procedures that improve over manual measurement procedures. The discipline is experiencing a transition, with many of its variables, e.g. conteo de yemas no brotadas, número de flores, cantidad de bayas, cantidad de racimos, número de plantas por claro, riqueza de poda, brotes totales, brotes de yemas francas, diámetro del tronco, longitud de entrenudos, longitud de 1er alambre descubierto y longitud del brote, entre tantos otros (Pellegrino et al., 2005; Intrigliolo and Castel, 2007; Reynolds and Heuvel, 2009; Matese and Di Gennaro, 2015; Ozdemir et al., 2017; Poni et al., 2018) are still being measured manually through visual inspection, resulting in large labor costs that limits the measurement campaigns to only small samples of data, that even with the use of statistical inference or spatial interpolation techniques impose a bound in the quality of the outcomes (Whelan et al., 1996; Borgogno-Mondino et al., 2018; Taylor et al., 2019). In some cases this is exacerbated by the need of experts for a proper measurement, such as the case of variables associated to the phenological stages of the plant such as bud swelling, bud burst, inflorescence, flowering, veraison, ripening of berries, among others (Lorenz et al., 1995); or by measurement procedure that requires the destruction of the part of the plant being measured, preventing any tracking of the variables overtime. Such is the case for the measurement of leaves área, bunch weight, berry weight and pruning weight (Kliewer and Dokoozlian, 2005).

Precision viticulture in general (Bramley, 2009), and computer vision algorithms in particular, has been growing in the last couple of decades, mainly for their potential for mitigating these limitations (Seng et al., 2018; Matese and Di Gennaro, 2015). These algorithms come along with a promise of an unprecedented boost in the production of vineyard information, with much expectations not only on possible improvements in the quality of the models' outcomes, but in its potential to produce better models by feeding all this information to big data algorithms.

In this work we contributed to this general endeavour with an algorithm

for measuring variables related to one specific part of the plant: the bud; an organ of major importance for being the grow point of the fruits, containing within all the productive potential of the plant (May, 2000). Our contribution of autonomous bud detection not only enables the autonomous measurement of all bud related variables currently measured by agronomists (see Table 1 for a non-exhaustive list of bud related variables); but has the potential to enable the measurement of novel, yet important variable that are currently impossible to be measured manually. One example is the total sunlight captured by the buds, that depends on the manually unfeasible task of determining the exact location of buds in 3D space. Although the present work focuses on 2D detection, it could be easily upgraded to 3D by, for instance, integrating the 2D detection in the workflow proposed by Díaz et al. (2018) (c.f. Section 1.1 for some more details on this workflow).

Table 1 shows a non-exhaustive list of the most important bud related variables currently measured by vineyard managers (Sánchez and Dokoozlian, 2005; Noyce et al., 2016; Collins et al., 2020), accompanied by an assessment of the extent to which detection contributes in their measurement. The right-most column indicates what information beyond detection is necessary to complete the measurement, while the middle columns labeled (i), (ii), and (iii) indicates the details of what specific aspects of the detection is required for that variable, whether it requires a good *segmentation*, i.e., the discrimination of which pixels in the scene correspond to buds and which ones correspond to the background (no-bud); *individualization*, i.e., discrimination of bud pixels as belonging to different buds; or *localization*, i.e., the localization of the bud within the scene; respectively. For instance, tomemos por caso la variable *buds number*. De ser posible individualizar correctamente las detecciones, the buds number se corresponde directamente con el conteo de detecciones. Por el contrario, para *bud type classification*, además de la individualización, la segmentación de la parte de la imagen correspondiente a la yema es necesaria para poder así alimentar a un clasificador con la información visual relevante, minimizando el ruido producto de pixeles del background. Por último, para medir la *incidence of sunlight on the bud*, no es necesaria la segmentación, sino tan solo una buena localización de la yema, además de la leaves 3D superficial geometry.

Variable/Aplicación	(i)	(ii)	(iii)	
Buds number		x		none
Bud area	x	x		none
Bud type classification	x	x		plant structure (trunk and canes)
Bud development stage	x	x		classifier over bud mask
Length between knots (by buds detection)		x	x	plant structure (trunk and canes)
Bud volume				3D reconstruction
Bud development monitoring	x	x	x	
Incidence of sunlight on the bud		x	x	3D reconstruction, leaves 3D superficial geometry

Table 1: Lista (no exhaustiva) de variables asociadas a las yemas, acompañadas de las sub-operaciones detección requeridas para su medición: (i) segmentación; (ii) individualización; y (iii) localización.

81 A good detector, therefore, should be evaluated on all three aspects of seg-
82 mentation, individualization and localization. This is easy for our detector as
83 its implementation first produces a segmentation mask, which is then post-
84 processed to produce the individualization and localization. Los detalles de este
85 enfoque se detallan en la Sección ???. El análisis de los resultados de detección
86 presentado en la Sección 3 muestra que este enfoque resulta superior a los
87 algoritmos del estado del arte para la detección de yemas de vid, además de que
88 demuestran ser suficientes para poder medir con buena calidad todas las vari-
89 ables de la Tabla (en algunos casos acompañadas de otros procesos complejos
90 como ser la construcción de un clasificador, por ejemplo). En la Sección 3.2.3
91 se discuten el alcance y las limitaciones de los resultados obtenidos para la
92 detección de yemas como también los futuros trabajos y posibles mejoras. Fi-
93 nalmente en la Sección 4 se presentan las conclusiones más importantes.

94 1.1. Related work

95 En la literatura se pueden encontrar una gran variedad de trabajos que
96 emplean algoritmos de computer vision y machine learning para adquirir infor-
97 mación sobre los viñedos (Seng et al., 2018), como ser berry and bunch detection
98 (Nuske et al., 2011), fruit size and weight estimation (Tardaguila et al., 2012),
99 leaf area indices and yield estimation (Diago et al., 2012), plant phenotyp-
100 ing (Herzog et al., 2014a,b), autonomous selective spraying (Berenstein et al.,
101 2010), y más (Tardaguila et al., 2012; Whalley and Shanmuganathan, 2013).
102 Entre los algoritmos de computer que se destacan en los últimos años, the *arti-*

103 *ficial neural networks* han despertado gran interés en la industria para llevar a
 104 cabo diversas tareas de reconocimiento visual (Hirano et al., 2006; Kahng et al.,
 105 2017; Tilgner et al., 2019). Particularmente las *Convolutional Neural Networks*
 106 (CNNs) se han convertido en el enfoque dominante de machine learning para el
 107 reconocimiento visual de objetos (Ning et al., 2017). Dos estudios recientes han
 108 aplicado exitosamente técnicas de reconocimiento visual basado en *deep learning*
 109 *networks* para identificar variables vitícolas que permitan estimar la producción
 110 en viñedos. Uno de ellos Grimm et al. (2019) utiliza una FCN para realizar
 111 segmentación de órganos de la planta de vid como los young shoots, pedicels,
 112 flower, buds or grapes. El segundo Rudolph et al. (2018) utiliza imágenes de
 113 vid en condiciones de campo que son segmentadas utilizando una CNN para
 114 detectar inflorescences y sobre esas regiones segmentadas se aplica el algoritmo
 115 circle Hough Transform para detectar las flowers buds.

116 Varios trabajos apuntan tanto a detectar como a localizar buds en diferentes
 117 tipos de cultivos mediante sistemas de reconocimiento visual autónomo. For
 118 instance Tarry et al. (2014) presents an integrated system for chrysanthemum
 119 bud detection that can be used to automate labour intensive tasks in floriculture
 120 greenhouses. More recently Zhao et al. (2018) presents a system of computer
 121 vision that is used to identify the internodes and buds of stalk crops. Según
 122 nuestro conocimiento y el mejor de nuestros esfuerzos de búsqueda, existen
 123 al menos cuatro trabajos que abordan el problema de la detección de yemas
 124 específicamente de la vid mediante sistemas de reconocimiento visual autónomo.
 125 Los trabajos presentados por Xu et al. (2014), Herzog et al. (2014b) y Pérez
 126 et al. (2017) aplican diferentes técnicas para realizar detección 2D en imágenes
 127 que involucra diferentes algoritmos de computer y machine learning. Además,
 128 Díaz et al. (2018) introduce un workflow para localizar yemas en el espacio 3D.
 129 A continuación se presentan los detalles más relevante de cada uno.

130 El trabajo de Xu et al. (2014), presenta un algoritmo de detección de yemas
 131 utilizando imágenes RGB capturadas indoor y condiciones controladas de ilumi-
 132 nación y fondo. Específicamente para establecer un groundwork para un sistema
 133 de podado autónomo en invierno. Los autores aplican un filtro por umbral para
 134 discriminar el fondo del esqueleto de la planta, resultando en una imagen bi-
 135 naria. Asumen que la forma de las yemas son similares a esquinas y aplican

136 el algoritmo *Harris corner detector* sobre la imagen binaria para detectarlas.
137 Este proceso obtiene un recall de 0.702, es decir el 70.2% de la yemas fueron
138 detectadas.

139 El trabajo de [Herzog et al. \(2014b\)](#) presenta tres métodos para la detección
140 de yemas. Todos los métodos utilizados se caracterizan por ser semi-automáticos
141 y requieren intervención humana para validar la calidad de los resultados. El
142 mejor resultado se obtiene utilizando una imagen RGB con un fondo artificial de
143 color negro y corresponde a un recall de 94%. Los autores argumentan que este
144 recall es suficiente para satisfacer el problema de fenotipado de plantas de vid.
145 También discuten que estos buenos resultados pueden explicarse debido al color
146 verde particular y la morfología de las yemas ya brotadas de aproximadamente
147 2cm.

148 En [Pérez et al. \(2017\)](#), presenta un enfoque para la clasificación de imágenes
149 de yemas en invierno, mediante un enfoque que emplea *SVM* como clasificador
150 y *Bag of Features* para computar descriptores visuales. Reportan un recall supe-
151 rior a 90% y una precision de 86% cuando se clasifican imágenes que contienen
152 al menos el 60% de una yema y una proporción del 20-80% de pixeles yema
153 vs pixeles no-yema. Argumentan que este clasificador puede ser utilizados en
154 algoritmos para localización 2D del tipo *sliding windows* debido a la robustez
155 ante la variación en tamaño y posición de la ventana. Es esta idea justamente
156 la que se ha reproducido en el presente trabajo para implementar el enfoque de
157 línea base basado en sliding windows y clasificador de patches.

158 Finalmente, en [Díaz et al. \(2018\)](#) se introduce un workflow para localización
159 de yemas en el espacio 3D. El workflow consta de 5 etapas. La primera real-
160 iza una reconstrucción a partir de varias imágenes RGB de una nube 3D de
161 puntos correspondientes a la estructura de la planta de vid. La segunda etapa
162 aplica un metodo de deteccion 2D utilizando una técnica de sliding window y
163 clasificación de patches. La etapa siguiente utiliza un esquema de votos para
164 clasificar cada punto de la nube como yema o no yema. La cuarta etapa aplica
165 el algoritmo de clustering *DBSCAN* para agrupar puntos de la nube que corre-
166 sponden a una yema. Finalmente en la quinta etapa se realiza la localización,
167 obteniendo las coordenadas del centro de masa de cada cluster de puntos 3D.
168 Reportan un recall de 45% con una precision de 100% y un error de localización

169 de aproximadamente 1.5cm , ó 3 diámetros de yema.

170 Si bien estos trabajos representan un gran avance en relación a la prob-
171 lemática de detección y localización de yemas, todavía sufren al menos una de
172 las siguientes limitaciones: (i) uso de fondo artificial en exteriores; (ii) ilumi-
173 nación controlada en interiores; (iii) necesidad de interacción con el usuario;
174 (iv) detección de yemas en etapas de desarrollo muy avanzado; y (v) bajo re-
175 call de detección/clasificación de yemas. Estas limitaciones representan una
176 importante barrera para el desarrollo efectivo de herramientas de medición de
177 variables asociadas a las yemas.

178 2. Materials and Methods

179 In this section we describe the main contribution of this work, the deep learn-
180 ing setup for the detection of grapevine buds in 2D images of vine plants cap-
181 tured in natural conditions. We start in the following subsection 2.1.1 with de-
182 tails on the *encoder-decoder* transfer learning architecture and the pre-training
183 chosen for its encoder; followed by subsection 2.1.2 describing our design of the
184 *scanning windows* detection procedure based on the state-of-the-art third-party
185 bud image classifier of Pérez et al. (2017), used as the strongest found com-
186 petitor to our proposed detection. We then proceed in subsection 2.2 with a
187 description of collection of the images used for training both the deep learning
188 and scanning windows models with details on the procedure used for its capture;
189 and conclude with subsection 2.3 with details on the procedure and parameters
190 for training of both models.

191 Como se describió en la introducción, el enfoque propone el uso de algoritmos
192 de visión computacional para: (i) *segmentar* las yemas *clasificando* cuales píxeles
193 de la escena corresponden a yema y cuales píxeles corresponden al background
194 (no-yema), (ii) *individualizar* las yemas distinguiendo entre aquellos pixeles que
195 pertenecen a diferentes yemas en la escena observada, y (iii) *localizar* cada yema
196 en la escena. Para la operación de segmentación, i.e., clasificación de pixeles, se
197 toma como base la FCN introducida en (Long et al., 2015), y se entrena para
198 el problema específico de segmentación de yemas de vid (ver sección 2.1.1).
199 La FCN resultante devuelve un mapa de probabilidad de igual escala que la
200 imagen original, donde el valor de un píxel representa la probabilidad de que

201 el píxel correspondiente en la imagen de entrada pertenezca a una yema. Para
 202 obtener una máscara binaria se aplica a cada píxel un umbral de clasificación τ ,
 203 clasificando al píxel como yema (no-yema) si su probabilidad es mayor (menor)
 204 a τ . Para individualizar las yemas se toma esta máscara binaria y se realiza un
 205 post-procesamiento para determinar que dos píxeles yema corresponden a una
 206 misma yema siempre y cuando pertenezcan a un mismo componente conectado,
 207 i.e., si los une alguna secuencia de píxeles yema contiguos. Finalmente, para
 208 la localización de objetos existen diversas alternativas entre las que encuentran
 209 *bounding box*, *pixel-wise segmentation*, *contorno* y *centro de masa del objeto*
 210 (Lampert et al., 2008). En este trabajo se tomó la última, eligiendo localizar a
 211 las yemas por el *centro de masa* de su componente conectado.

212 Los resultados de detección alcanzados por este enfoque son contrastados
 213 con el método de detección de yemas introducido en Pérez et al. (2017). En
 214 este trabajo los autores proponen el uso de *sliding windows* para subdividir
 215 la imagen en un conjunto de *patches* o regiones más pequeñas (Pérez et al.,
 216 2017), y luego determina si cierto patch contiene o no una yema usando un
 217 clasificador de imágenes construido con el algoritmo *Support Vector Machine*
 218 (Vapnik, 2013). Para poder contrastar ambos enfoques cada uno recibe el mismo
 219 tipo de entrada, i.e. una imagen de una escena vitícola, y producen las mismas
 220 salidas, i.e. una máscara binaria del mismo tamaño que la imagen original cuyos
 221 píxeles positivos representan los píxeles del tipo yema, junto a las coordenadas
 222 (X,Y) de la localización de estas yemas. A continuación se dan los detalles de
 223 cada implementación.

224 2.1. Models

225 2.1.1. Fully Convolutional Network with MobileNet (FCN-MN)

226 Como clasificador de píxeles se utilizaron las tres versiones 32s, 16s y 8s
 227 de las FCN introducidas originalmente por Long et al. (2015), por haber sido
 228 utilizadas con excelentes resultados en muchas aplicaciones de segmentación de
 229 imágenes Litjens et al. (2017); Garcia-Garcia et al. (2018); Kaymak and Uçar
 230 (2019). Estas redes presentan arquitecturas características con dos partes bien
 231 distinguibles: *encoder* y *decoder* (ver 1). El encoder consiste en una CNN
 232 que realiza un *downsampling* de una imagen de entrada en un conjunto de fea-
 233 tures mediante operaciones de convolución, para producir un conjunto de *feature*

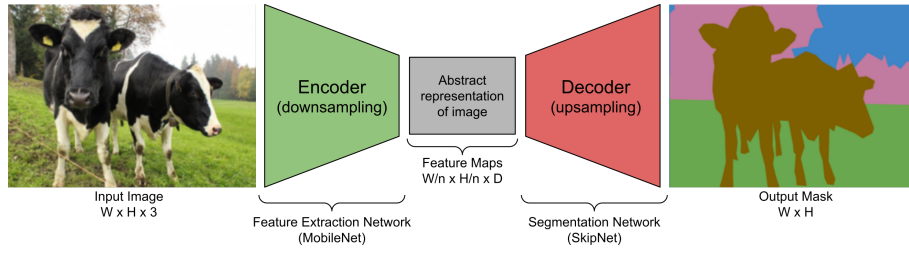


Figure 1: Esquema de la arquitectura de red FCN-MN utilizada en este trabajo, basada en la FCN propuesta por [Shelhamer et al. \(2017\)](#), reemplazando su encoder de extracción de features por las redes MobileNet [Howard et al. \(2017\)](#), lo que produce features maps con un factor de downsampling n . Como decoder para la producción del mapa de segmentación se utiliza la red SkipNet [Siam et al. \(2018\)](#), implementando las variantes 32s, 16s y 8s.

234 *maps*, i.e. una representación abstracta de la imagen que captura información
 235 semántica y contextual, pero que descarta información espacial de grano fino.
 236 Estas operaciones reducen las dimensiones espaciales de la imagen a medida que
 237 se avanza más profundo en la red, resultando en feature maps de tamaño $1/n$
 238 del tamaño de la imagen de entrada, donde n es el factor de downsampling. El
 239 decoder es una subred de *upsampling*, que toma el conjunto de feature maps de
 240 baja resolución y los proyecta al espacio de píxeles, aumentando la resolución
 241 para producir una máscara de segmentación (o clasificación densa de píxeles)
 242 con las mismas dimensiones de la imagen de entrada. Esta operación se imple-
 243 menta como una red de transposed convolutions con parámetros entrenables,
 244 también conocidas como upsample convolutions [Shelhamer et al. \(2017\)](#).

245 Por otra parte, para refinar la calidad de la segmentación, se suelen utilizar
 246 conexiones que sobrepasan al menos una capa de la red, llamadas *skip connec-*
 247 *tions*. Éstas se utilizan para transferir información espacial local desde las capas
 248 internas del encoder directamente al decoder. En general, estas conexiones mejo-
 249 ran los resultados de segmentación, ya que mitigan la pérdida de información
 250 espacial permitiendo al decoder incorporar información de feature maps inter-
 251 nos, aunque su impacto puede variar según la skip architecture que se proponga.
 252 En [Long et al. \(2015\)](#) se proponen tres skip architectures: la 32s sin información
 253 de capas internas del encoder; la 16s que suma información espacial de capas
 254 profundas del encoder; y la 8s, que suma información espacial de capas profun-
 255 das y menos profundas del encoder. Los detalles de estas arquitecturas quedan
 256 fuera del alcance de este trabajo, pero pueden consultarse en [Long et al. \(2015\)](#)

257 y [Shelhamer et al. \(2017\)](#). Dado que los resultados reportados en la literatura
258 no son concluyentes respecto a que arquitectura es mejor [Long et al. \(2015\)](#);
259 [Shelhamer et al. \(2017\)](#), en este trabajo se consideran las tres alternativas.

260 A pesar de haber alcanzado excelentes resultados en la práctica, estas ar-
261 quitecturas conllevan una importante carga de recursos computacionales. Con
262 esto en mente, en este trabajo se reemplazó el encoder VGG [Simonyan and](#)
263 [Zisserman \(2015\)](#) propuesto originalmente por Long para las FCN, por la red
264 MobileNet [Howard et al. \(2017\)](#), una red que se destaca por tener tan solo 4.2
265 millones de parámetros frente a los 138 millones de parámetros de VGG, per-
266 mitiendo que el proceso de entrenamiento y testeo sea considerablemente más
267 rápido, con requerimientos de memoria muy inferiores, pero manteniendo la per-
268 formance. El uso de MobileNet como encoder en las FCN de [Long et al. \(2015\)](#)
269 no es novedoso, sino que ha sido ya propuesto para la arquitectura 8s por [Siam](#)
270 [et al. \(2018\)](#) en su arquitectura SkipNet. Técnicamente, la propuesta de [Siam](#)
271 [et al. \(2018\)](#) es sumamente sencilla, por lo que nos atrevemos aquí a extenderla
272 a las arquitecturas 16s y 32s propuestas originalmente por ([Long et al., 2015](#)).
273 Debido a estos cambios es que nos referimos a estas redes como FCN-MN de
274 aquí a lo que resta del paper.

275 2.1.2. *Sliding Windows y Clasificador de patches (SW-C)*

276 En esta sección se describe el enfoque propuesto por [Pérez et al. \(2017\)](#)
277 para clasificación de imágenes de yema y una implementación del mismo para
278 detección basada en sliding windows descrita en el trabajo original.

279 Este enfoque opera en tres pasos: (i) aplica el algoritmo de sliding windows
280 sobre una imagen para extraer patches (sub-imágenes o regiones rectangulares);
281 (ii) clasifica (todos los píxeles de) cada patch en yema o no-yema mediante
282 el algoritmo presentado en [Pérez et al. \(2017\)](#); y (iii) produce la máscara de
283 segmentación final mediante un esquema de votación. A continuación se dan
284 los detalles de cada paso.

285 Las técnicas sliding windows comprenden una familia de algoritmos amplia-
286 mente utilizados en el pasado como parte de diversos enfoques para localización
287 de objetos con bounding boxes ([Divvala et al., 2009](#); [Wang et al., 2009](#); [Chum](#)
288 [and Zisserman, 2007](#); [Ferrari et al., 2007](#); [Dalal and Triggs, 2005](#); [Rowley et al.,](#)
289 [1996](#)). En estos algoritmos, cada imagen es escaneada densamente desde un ex-

290 tremo de la imagen (e.g. esquina superior izquierda) hasta el otro extremo (e.g.
 291 esquina inferior derecha) mediante una ventana deslizante rectangular en difer-
 292 entes escalas y diferentes desplazamientos, extrayendo sub-imágenes o patches
 293 de la imagen original. En este trabajo, se definen 10 tamaños de ventana de
 294 igual alto y ancho, a saber 100, 200, 300, 400, 500, 600, 700, 800, 900 y 1000
 295 píxeles, con un desplazamiento horizontal del 50% el ancho de la ventana y un
 296 desplazamiento vertical del 50% el alto de la ventana, lo que produce una super-
 297 posición del 50% entre parches contiguos. Estos valores se eligen sobre la base
 298 del análisis de robustez del clasificador que presenta Pérez et al. (2017) para
 299 la geometría de la ventana. Este análisis muestra que el clasificador (explicado
 300 en la sección 2.3.2) es robusto para los patches que contienen al menos 60% de
 301 los píxeles de una yema, y estos deben cubrir al menos el 20% del patch. Si
 302 consideramos los casos extremos, i.e. el diámetro de yema más pequeño 100px y
 303 el más grande 1600px, tamaños de ventana de 100px y 1000px podrían contener
 304 al menos el 60

305 El segundo paso de este enfoque consiste en determinar si un patch es de
 306 clase yema o no-yema. El clasificador de Pérez et al. (2017) toma los patches
 307 producidos por el sliding windows y para cada uno realiza las siguiente opera-
 308 ciones: (i) computa features visuales de bajo nivel mediante el algoritmo *Scale*
 309 *Invariant Feature Transform* (SIFT) Lowe (2004); (ii) construye un descriptor
 310 de alto nivel para cada patch empleando el algoritmo *Bag of Features* (BoF)
 311 Csurka et al. (2004) sobre los features SIFT del paso anterior; y (iii) determina
 312 la clase de cada patch usando el descriptor BoF sobre un clasificador construido
 313 mediante el algoritmo *Support Vectors Machine* Vapnik (2013). Los detalles del
 314 entrenamiento de este clasificador se posponen hasta la sección 2.3.2 (Entre-
 315 namiento SW-C).

316 Finalmente, el tercer paso del enfoque consiste en construir la máscara bina-
 317 ria donde se encuentran etiquetados los píxeles que pertenecen a la clase yema
 318 y no-yema. Esta máscara es construida a través de un esquema de votación
 319 donde cada píxel suma un voto por cada patch que lo contiene clasificado como
 320 yema, el cual podría ser de un máximo de 4 para algunos píxeles debido a que el
 321 deslizamiento propuesto entre patches presenta solapamiento tanto horizontal
 322 como vertical. Luego, se establece un umbral de votos mínimos ν que puede

323 tomar los valores del 1 al 4, de tal manera que los píxeles con una cantidad de
324 votos igual o mayor a ν son clasificados como yema, caso contrario se clasifican
325 como no-yema.

326 2.2. Colección de imágenes

327 La colección de imágenes utilizada en este estudio es la misma colección
328 utilizada originalmente en Pérez et al. (2017), el cual se ha descargado de la URL
329 indicada por los autores <http://dharma.frm.utn.edu.ar/vise/bc>. La colección
330 completo está compuesta por 760 imágenes capturadas en condiciones natural
331 de campo, en invierno. Sin embargo en este trabajo solo se tomaron las 698
332 imágenes que contienen exactamente una yema. Cada imagen está acompañada
333 del ground truth, es decir una máscara con la segmentación manual de la yema.
334 Estas imágenes y sus máscaras fueron empleadas durante el entrenamiento y
335 evaluación de los modelos de detección. Para esto, el corpus de imágenes se
336 separó en dos subconjuntos disjuntos: el *trainset* con el 80% de las imágenes y
337 el *testset* con el restante 20%. Esto resultó en un trainset de 558 imágenes y
338 un testset de 140 imágenes, ambos con sus respectivas máscaras ground truth.
339 De esta manera, los dos enfoques propuestos utilizan exactamente las mismas
340 558 imágenes durante el entrenamiento, y las mismas 140 imágenes durante la
341 evaluación.

342 2.3. Entrenamiento de los modelos

343 En esta sección se dan los detalles del proceso de entrenamiento para cada
344 enfoque empleando las 558 imágenes del trainset.

345 2.3.1. Entrenamiento enfoque FCN-MN.

346 Para el entrenamiento de este enfoque se utilizaron las 558 imágenes reser-
347 vadas para este propósito, las mismas que se usaron para el entrenamiento del
348 enfoque anterior. Estas imágenes presentan diferentes resoluciones, sin embargo
349 las FCN-MN requieren una entrada de tamaño fijo. Por esto, todas las imágenes
350 (incluida sus máscaras) fueron escaladas a una resolución de 1024×1024 píxeles
351 usando un método de interpolación bilinear (Han, 2013). Además, para las
352 imágenes del trainset se realizó un scaling en los valores de intensidad RGB de
353 los píxeles de $[0, 255]$ a $[-1, 1]$.

354 Dado que la cantidad de imágenes en el trainset se considera escasa, para
 355 lograr un entrenamiento robusto se emplearon dos técnicas ampliamente uti-
 356 lizadas en la práctica: *transfer learning* Pan and Yang (2009) y *data augmen-*
 357 *tation* Shorten and Khoshgoftaar (2019). El proceso de transfer learning se
 358 realizó de la siguiente manera: (i) se implementa la red MobileNet original
 359 propuesta en Howard et al. (2017); (ii) se inicializa la red con los parámetros
 360 pre-entrenados sobre el dataset de benchmark ImageNet Kornblith et al. (2019);
 361 (iii) se reemplaza la capa de clasificación multiclase de MobileNet por una capa
 362 de clasificación binaria; (iv) se entrena la red como un clasificador de patches
 363 yema y no-yema de forma análoga al entrenamiento de SVM, empleando el
 364 trainset de patches balanceado luego de escalar todas sus imágenes a 224×224
 365 píxeles; y (v) se toman los parámetros obtenidos en este pipeline para inicializar
 366 el encoder de nuestra FCN-MN, introducido en la sección 2.1.1. El proceso de
 367 data augmentation se aplicó on the fly durante el entrenamiento, i.e. en la
 368 medida que el proceso requería nuevas imágenes. Por cada imagen del tra-
 369 iset se generaron 200 nuevas imágenes (111600 en total) aplicando simultáneamente
 370 las siguientes siete operaciones, donde sus valores se tomaron de forma aleato-
 371 ria con probabilidad uniforme: *rotación* de hasta 45° ; *traslación horizontal* de
 372 hasta 40%; *traslación vertical* de hasta 40%; *shear* de hasta 10%; *Zoom* de
 373 hasta 30%; *flip horizontal*; y *flip vertical*. En general, para el entrenamiento
 374 de una FCN-MN se requiere especificar el *método de optimización* y el valor
 375 de *dropout*, dos parámetros típicamente definidos por el usuario. En este tra-
 376 bajo, los métodos de optimización que se tuvieron en cuenta fueron: *Adam* con
 377 parámetros *LearningRate* = 0.001, *beta1* = 0.9 y *beta2* = 0.999; *RMSProp* con
 378 parámetros *LearningRate* = 0.001 y *rho* = 0.9; y *Stochastic Gradient Descent*
 379 con parámetros *LearningRate* = 0.0001 y *Momentum* = 0.9. Para el caso de
 380 dropout se definieron los valores 0.5 y 0.001. Estos valores fueron preselecciona-
 381 dos por experimentaciones preliminares que no se discuten aquí.

382 La mejor combinación entre método de optimización y valor de dropout se
 383 estableció en tiempo de entrenamiento sobre un conjunto de validación, uti-
 384 lizando el enfoque *4-fold cross validation* por 60 epochs, variando sobre los tres
 385 métodos de optimización y los dos valores de dropout. Los valores seleccionados
 386 fueron aquellos que maximizan el promedio de la Jaccard's *Intersection-over-*

Optimizer	Mean IoU	
	Dropout = 0.001	Dropout = 0.5
RMSprop	<u>0.44253</u>	0.3117
Adam	0.240277	0.315714
SGD	0.000886	0.00151

Table 2: Promedio de IoU sobre las 3 variantes
para cada combinación de parámetros.

387 *Union* (IoU) (Jaccard, 1912), en los 4-folds sobre las 3 variantes, siendo IoU
388 una medida de evaluación típica de problemas de segmentación (ver sección
389 3.1.2). Observamos en la Tabla 2 que la combinación de parámetros con la que
390 se alcanza mayor IoU promedio es RMSProp con dropout de 0.001.

391 Finalmente se procedió a entrenar las 3 variantes con RMSProp como método
392 de optimización y un valor de dropout de 0.001 sobre el conjunto de entre-
393 namiento completo por 200 epochs.

394 2.3.2. Entrenamiento enfoque SW-C

395 La etapa de entrenamiento para este enfoque se realiza de la misma manera
396 que para el workflow original propuesto en Pérez et al. (2017). Esto implica
397 entrenar un clasificador binario para que aprenda el concepto de yema versus
398 no-yema a partir de un corpus de patches rectangulares que contienen o no
399 una yema. Durante el entrenamiento, los patches yema deben ser regiones que
400 circunscriben perfectamente la yema (ver 2). Los patches no-yema deben ser
401 regiones que no contienen ni un solo píxel de yema (ver 2). Por lo tanto, para
402 construir el corpus de patches, se procesaron las 558 imágenes y sus máscaras
403 siguiendo el mismo protocolo que en Pérez et al. (2017), obteniendo un total de
404 558 patches que circunscriben a cada yema (existe una por imagen) y más de
405 25000 patches no-yema (el área no-yema es mucho mayor al área que ocupa una
406 yema en la imagen). El tamaño de estos patches es variable, con resoluciones
407 entre 0.1 y 2.6 megapíxeles aproximadamente (patches de 100×100 a 1600×1600
408 píxeles).

409 A partir de este corpus de patches, se creó un trainset de patches balanceado,
410 i.e. con 558 patches de cada clase, donde los patches no-yema fueron tomados al



Figure 2: Collection of patches used in this work. The first row corresponds to bud patches. The second and third rows correspond to the non-bud patches.

411 azar entre miles de patches. El entrenamiento se realizó tal como se detalla en
 412 el pipeline propuesto en Pérez et al. (2017): (i) se extrajeron descriptores SIFT
 413 todos los patches del trainset; (ii) se aplicó BoF con tamaño de vocabulario
 414 igual a 25, dado que fue el modelo con mejores resultados según los autores; y
 415 (iii) se entrenó el clasificador SVM sobre los descriptores BoF de cada patch,
 416 empleando un kernel *Radial Basis Function*, donde el valor de los parámetros γ
 417 y C se estableció mediante un 5-fold cross-validation sobre los mismos rangos
 418 de valores, i.e. $\gamma = \{2^{-14}, 2^{-13}, \dots, 2^{-7}\}$ y $C = \{2^5, 2^6, \dots, 2^{14}\}$.

419 3. Experimental results

420 In this section we present a systematic evaluation of the quality our pro-
 421 posed procedure FCN-MN for bud detections quality, which, according to the
 422 discussion in the introduction, can be decomposed on the three aspects that
 423 impact on the relevant bud related variables listed in Table ??: *segmentation*,
 424 *individualization*, and *localization*.

425 For that, we start in the following subsection by presenting metrics that
 426 quantify the quality for these aspects, followed by the results subsection 3 that
 427 presents details on the metric values obtained for different experiments over the
 428 test set of images.

429 3.1. Performance metrics

430 3.1.1. Individualization metrics

431 Individualization of buds, in both FCN-MN and SW, is the result of two
432 steps: (i) the thresholding of the algorithm’s output mask into a *binary mask*,
433 keeping all pixels of ν the probabilistic mask output by FCN-MN with values
434 higher than τ and keeping all pixels belong to at least ν patches rendered positive
435 by SW, and (ii) the association of each *connected component* of the binary mask
436 to exactly one (detected) bud.

437 An incorrect individualization is thus the result of incorrect matching of
438 detected components with actual buds in the image. This matching can get
439 very complicated when there is an unknown number of true buds in the scene
440 as can be seen by the large amount of possible detection metrics defined in [Oguz
441 et al. \(2017\)](#). To simplify the analysis our image corpus contains a single bud
442 per image, avoiding the need of all metrics that report the confusing situation of
443 a component overlapping more than one true bud. This results in the following
444 simplified list of possible metrics:

- 445 • **Correct Detection** (CD) is the best case, and counts all images in the
446 test corpus for which the detected binary mask presents a single connected
447 component, and this connected component overlaps with the true bud of
448 the image. This would correspond with a *true positive* situation.
- 449 • **Split** (S) occurs when there is more than one detection per bud, which
450 happens when the mask contains multiple connected components, all of
451 which overlaps the true bud. This metric counts the total number of such
452 components over all images in the test corpus.
- 453 • **False Alarm** (S), is equivalent to a *false positive* situation, and corre-
454 sponds to connected components not overlapping with the true bud. As
455 for splits, it counts, for each image, the number of such components.
- 456 • **Detection Failure** (DF), is equivalent to a *false negative* situation, when
457 the detection mask presents no connected components. It counts one each
458 image satisfying this condition.

459 All four of these cases are mutually exclusive, that is, no image can satisfy
 460 any two (or more) of these definitions simultaneously. To quantify the indi-
 461 vidualization quality one could simply report these quantities counted over the
 462 test set, with the best case consisting in a CD value equal to the cardinality
 463 of this set. However, determining the overall individualization quality from the
 464 analysis of 4 quantities can get rather complicated. One alternative is report-
 465 ing the well-known precision and recall, denoted P_D and R_D and referred to as
 466 *detection-precision* and *detection-recall* to distinguish them from the segmenta-
 467 tion precision and recall defined later below. For that, we have to address first
 468 the fact that we have two differing true positive counts: CD and S . We solve
 469 this by first counting as true positives not only the CD type of images, but
 470 the S ones, i.e., we count as one true positive any image with either a correct
 471 detection or a split case, resulting in:

$$P_D = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} = \frac{CD + S}{CD + S + FA} \quad (1)$$

$$R_D = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} = \frac{CD + S}{CD + S + DF}, \quad (2)$$

472 and then account for the split type of errors by explicitly reporting S .

473 Given these quantities we also report the *F1-measure* computed as their
 474 harmonic average:

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

475 3.1.2. Segmentation metrics

476 Individualization metrics, although informative, relies on the overlap be-
 477 tween the detected and true buds, regardless of how minimal the overlap. This
 478 could miss several possible pixelwise detection errors, resulting in rather coarse
 479 comparisons between competing detection algorithms. For instance, a correct
 480 detection could present a very small overlap with the true bud, with many or
 481 even a majority of the true bud's pixels missing (i.e., several *false negatives* pix-
 482 els), or could be erroneously reporting several pixels as bud pixels (i.e., several
 483 *false positives* pixels). Clearly, the best case scenario would be a case of correct
 484 detection with no false negative or positive pixels, that visually would corre-
 485 spond to a perfect overlap of the detected connected component and the true

486 bud. Similarly, a pixel wise comparison of the masks could help assess the qual-
 487 ity of the splits. The best split, for instance, would be one completely enclosed
 488 within the true mask, i.e., with none of its connected components presenting
 489 false positive pixels; while covering as much of the true bud mask as possible,
 490 i.e., presenting just enough false negatives to disconnect its components. Fi-
 491 nally, a false alarm case, clearly presenting only false positive pixels, could be
 492 further assessed by the number of (false positive) pixels in its components.

493 The community has proposed several metrics to quantify segmentation er-
 494 rors. The most obvious ones are those that report the *fraction* of the whole
 495 image corresponding to *true positive* pixels, denoted TPF ; *false positive* pixels,
 496 denoted FPF ; and *false negative* pixels, denoted FNF . As for the individ-
 497 ualization metrics, one can simplify the analysis by considering the pixelwise
 498 precision and recall, denoted P_S and R_S and referred to as *segmentation preci-*
 499 *sion* and *segmentation recall*, defined formally as:

$$\begin{aligned} P_S &= TPF / (TPF + FPF) \\ R_S &= TPF / (TPF + FNF), \end{aligned}$$

$$2 \times \text{precision} \times \text{recall} / (\text{precision} + \text{recall}), \quad (3)$$

500 proposed independently by [Dice \(1945\)](#), thus usually referred to as the
 501 *Dice measure*. A common alternative to the Dice measure is the Jaccard's
 502 *intersection-over-union* ([Jaccard, 1912](#)), equivalent to $TPF / (TPF + FPF +$
 503 $FNF)$.

504 With these metrics, one could quantify the refinements discussed in the
 505 first paragraph above, by simply applying them, not to the whole mask, but to
 506 the individual individualization cases. For instance, reporting the mean Dice
 507 measured over all correctly detected components; or, to refine the assessment of
 508 how bad is a split, one could report the mean Dice measure to all components
 509 of some split, or the mean Dice measure over all split components of all split
 510 images.

511 The case of false alarms is rather monotonous and not very informative, with
 512 zero precision and recall for all such components. Indeed, a pixelwise assessment

of the gravity of a false alarm requires a quantification of the number of false positive pixels. One could simply consider the *FPF*, the fraction of all the image pixels that are false positives. Instead, we considered a normalization against the size of the bud to be more informative, resulting in the *normalized area*, denoted *NA* and defined formally as *the total area of the component corresponding to its total number of pixels, normalized by the area of the true bud*.

3.1.3. Localization metrics

As a localization metric we propose the *normalized distance*, denoted *ND*, defined formally as *the distance between the center of mass of the component, to the center of mass of the true bud, divided by the diameter of the true bud (defined as the maximum distance between any two border points of the true bud)*.

3.2. Resultados Sistemáticos

We proceed now to assess the validity of our main hypothesis, namely, that FCN-MN is a better detector than its SW counterpart over each of the metrics defined in the previous section.

For a thorough comparison we considered several cases for each algorithm, training 27 FCN-MN detectors and 40 SW detectors over the training set of 558 images, one for each combination of their respective hyper-parameters. For FCN-MN these hyper-parameters are the three architectures 8s, 16s, and 32s, and the 9 values $\{0.1, 0.2, \dots, 0.9\}$ for the binarization threshold τ ; whereas for SW these hyper-parameters are the 10 patch sizes $\{100, 200, \dots, 1000\}$ and the 4 values $\{1, 2, 3, 4\}$ of the voting threshold ν .

Table 3 shows the results for the best detectors of each algorithm, reporting all performance metrics of the three aspects of detection: individualization, segmentation and localization. The first column shows the label of the selected detectors, with the subscript indicating the architecture and patch size for the case of FCN-MN and SW, respectively, while the superscript indicating the thresholds τ and ν , respectively.

The table includes all metrics defined in Section 3.1 required for a thorough comparison of FCN-MN against SW. First, we include four individualization metrics: detection precision P_D , detection recall R_D , the F1-measure $F1$, and

S (the total count of split components). For a thorough analysis of the segmen-
 tations we discriminated the segmentation metrics for the correctly detected,
 splitted and false alarms. For the detections, i.e., correctly detected and splits,
 we report segmentation precision, segmentation recall, and the Dice measure
 denoted in the table by P_S^{CD} , R_S^{CD} and $Dice^{CD}$ for the correctly detected, and
 P_S^S , R_S^S and $Dice^S$ for the splits. Each of the three correctly detected cells report
 the mean value of the measure computed for each correctly detected test image,
 i.e., each image with only one component overlapping the true bud, including
 the corresponding standard deviation in parenthesis. For the split group, the
 mean and standard deviation are computed over the measures computed only
 for the split images, i.e., over the images containing at least two components
 overlapping the true bud. Here, the segmentation metrics are computed over
 the union of all split components. For the false alarms we reported the mean
normalized area(NA), in this case computed individually for each false alarm
 component, reporting at each cell its mean over all false alarm components of
 all test images.

Finally, for localization the table reports the *normalized distance*(ND) only
 for false alarms, considering that correctly detected and splits, as they overlap
 the true bud, should be close enough to render it unnecessary further analysis.
 Instead, a false alarm can be arbitrarily far from the true bud. We thus report
 in the column ND the mean normalized distance of each false alarm connected
 component that appears in any test image.

The table is a summary, as it includes only a subset of all 27 FCN-MN
 cases, and a subset of all 40 SW cases. A detector was considered for inclusion
 in the table if, when compared to its counterparts of the same algorithm, it
 resulted in the higher value for at least one of the metrics. The corresponding
 cell was marked in bold in the table. For instance, the detectors $FCN-MN_{16s}^{0.8}$
 is included because its detection precision P_D of 97.7 is the largest among the
 detection precision of all 27 FCN-MN detectors. Similarly, the detectors SW_{1000}^1
 has been included because its precision $P_D = 67.0$ is the largest among all 40
 SW detectors.

The table shows a clear improvement of FCN-MN over SW. For all metrics it
 is the case that the best FCN-MN detector (bolded) improves (or ties) over the

best SW detector (bolded); represented in the table by underlying the one with better metric; with the exception of the two segmentation recalls (for correctly detected and splits) for which the SW case has a better (larger) mean, 98.8 versus 99.9 for correctly detected, and 74.7 versus 78.6 for the split case; and the total split count S , with the best case for FCN-MN being 1 and 0 for the best SW case. These improvements are not statistically significant, however, due to the large standard deviations of the FCN-MN cases, of 3.4 and 8.1, for the correctly detected and split cases, respectively, resulting in (statistically) overlapping values. In some cases the improvements of FCN-MN over SW are overwhelming. For instance, for the detection-precision, the correctly detected segmentation-precision, and the split segmentation-precision, the FCN-MN over SW improvements are 97.7 versus 67.0, 98.1 versus 46.5 and 99.9 versus 67.5, respectively. Also, for NA and ND the FCN-MN versus SW improvements are 0.04 versus 0.22, and 1.1 versus 6.0, respectively.

3.2.1. Detailed analysis of individualization metrics

Graphically one could expect a better combined analysis of the detection-precision and detection-recall than one could obtain by comparing the F1-measure. This is shown as a scatter plot in Figure [refig:detection-scatter-plot](#), a graphical representation of a non-summarized version of the second and third columns of Table [3](#). Each dot in the plot is located according to the detection-precision and detection-recall, and the colored black or white whether it corresponds to an FCN-MN or an SW detection model. The graph reinforces the clear and undisputed improvements of FCN-MN over SW already detected in the table, with similar detection-recalls but larger detection-precisions over the majority of scenarios, resulting in a larger area under the PR curve.

Detection-precision and detection-recall are computed over a combination of correctly detected and splitted components. To easily assess the impact of the split cases, we show in Figure [4](#) the S values, corresponding to the fifth column of a (non-summarized version of) Table [3](#) in the form of a histogram; with bins representing values of S , and the bars for that bin representing the proportion of models that resulted in that value of S . Black and white bars discriminate the cases for FCN-MN and SW, respectively. For instance, the first bin indicates that approximately 54% of the FCN-MN models and approximately 62% of the

Detector	P_D	R_D	$F1$	S	P_S^{CD}	R_S^{CD}	$Dice^{CD}$	P_S^S	R_S^S	$Dice^S$	NA	ND
FCN-MN _{8s} ^{0.5}	75.4	98.6	85.4	2	91.0 (11.3)	90.2 (11.7)	89.6 (10.3)	96.6 (2.2)	73.1 (17.6)	82.1 (10.2)	0.26 (0.69)	3.72 (4.64)
FCN-MN _{8s} ^{0.9}	90.1	97.1	93.5	8	98.1 (6.0)	68.3 (21.1)	77.9 (19.6)	98.7 (3.0)	57.4 (18.4)	70.8 (13.6)	0.24 (0.5)	3.8 (5.66)
FCN-MN _{16s} ^{0.1}	71.3	100	83.2	6	75.7 (13.1)	95.4 (14.7)	83.1 (13.5)	83.1 (8.9)	54.1 (21.9)	61.9 (17.5)	0.12 (0.44)	5.27 (6.53)
FCN-MN _{16s} ^{0.4}	87.0	96.4	91.5	1	87.7 (12.1)	89.8 (18.2)	87.0 (15.6)	96.7 (0.0)	37.0 (0.0)	53.5 (0.0)	0.04 (0.09)	3.8 (5.08)
FCN-MN _{16s} ^{0.6}	95.6	93.6	94.6	3	92.2 (8.7)	88.2 (13.3)	89.1 (10.7)	99.4 (0.6)	16.2 (10.6)	26.6 (16.8)	0.08 (0.11)	1.1 (0.65)
FCN-MN _{16s} ^{0.8}	97.7	92.1	94.9	4	95.8 (7.0)	81.6 (14.6)	87.0 (10.7)	99.7 (0.3)	34.2 (32.6)	43.9 (33.1)	0.1 (0.12)	1.28 (0.95)
FCN-MN _{16s} ^{0.9}	97.7	91.4	94.5	4	97.6 (5.6)	74.5 (16.5)	83.1 (12.8)	99.9 (0.1)	31.8 (27.9)	41.6 (34.0)	0.07 (0.11)	1.33 (0.9)
FCN-MN _{32s} ^{0.1}	35.4	100	52.2	8	67.4 (14.0)	98.8 (3.4)	79.1 (11.0)	86.0 (9.4)	73.4 (19.6)	77.1 (10.4)	0.14 (0.66)	4.62 (5.59)
FCN-MN _{32s} ^{0.2}	50.9	100	67.5	10	73.9 (13.6)	98.1 (3.8)	83.5 (10.1)	92.2 (5.4)	53.4 (25.8)	63.6 (19.3)	0.17 (0.55)	4.33 (6.17)
FCN-MN _{32s} ^{0.3}	49.8	100	66.5	10	79.1 (13.2)	95.5 (10.5)	85.2 (11.8)	88.5 (9.7)	61.0 (35.1)	65.8 (28.2)	0.1 (0.39)	3.68 (5.62)
FCN-MN _{32s} ^{0.6}	68.5	99.3	81.1	16	89.0 (11.5)	89.1 (11.3)	88.1 (9.6)	92.4 (7.7)	74.7 (28.1)	78.1 (24.0)	0.11 (0.3)	2.95 (4.36)
SW ₁₀₀ ¹	9.4	100	17.2	28	24.6 (17.7)	86.7 (19.5)	33.6 (15.1)	57.9 (28.2)	24.8 (16.8)	27.9 (13.8)	1.08 (3.2)	7.68 (6.02)
SW ₁₀₀ ³	14.6	93.1	25.3	40	42.4 (26.4)	56.8 (29.9)	39.9 (19.7)	55.5 (32.2)	24.8 (18.1)	26.0 (15.6)	0.31 (0.96)	6.45 (6.19)
SW ₁₀₀ ⁴	19.5	87.4	31.9	49	46.5 (29.3)	39.2 (28.9)	33.9 (21.1)	49.0 (29.0)	20.1 (13.7)	24.1 (14.0)	0.22 (0.57)	6.0 (6.56)
SW ₂₀₀ ¹	20.0	100	33.3	12	16.6 (12.5)	94.9 (13.5)	25.9 (14.2)	49.3 (26.4)	40.2 (17.4)	36.8 (11.9)	5.13 (19.3)	7.56 (5.35)
SW ₂₀₀ ³	26.0	98.6	41.1	19	29.9 (17.0)	74.7 (27.3)	38.5 (17.0)	67.5 (32.7)	16.5 (8.9)	24.2 (11.9)	1.69 (3.15)	8.94 (6.22)
SW ₃₀₀ ¹	26.9	100	42.4	2	13.7 (13.6)	97.0 (9.6)	21.6 (15.5)	55.0 (11.8)	48.1 (1.1)	50.8 (4.5)	7.79 (20.5)	6.83 (4.44)
SW ₄₀₀ ¹	32.7	100	49.3	2	10.5 (11.7)	98.7 (9.3)	17.2 (15.3)	42.6 (10.1)	61.9 (11.6)	50.4 (10.9)	11.59 (24.05)	7.12 (4.15)
SW ₄₀₀ ²	34.6	100	51.4	4	15.6 (15.1)	94.5 (13.3)	23.8 (15.6)	48.7 (27.6)	36.0 (4.6)	38.6 (13.1)	9.54 (26.13)	7.88 (4.89)
SW ₅₀₀ ¹	40.2	100	57.3	1	8.40 (9.7)	99.9 (4.9)	14.2 (13.8)	17.9 (0.0)	78.6 (0.0)	29.2 (0.0)	17.39 (30.07)	7.22 (4.04)
SW ₅₀₀ ²	38.6	100	55.7	1	13.5 (14.0)	95.2 (14.5)	21.0 (16.0)	35.2 (0.0)	45.9 (0.0)	39.8 (0.0)	17.19 (39.07)	7.56 (4.42)
SW ₆₀₀ ¹	43.5	100	60.6	0	6.9 (7.8)	98.5 (10.7)	12.0 (12.0)	nan (nan)	nan (nan)	nan (nan)	25.48 (48.45)	7.72 (4.3)
SW ₆₀₀ ²	41.7	100	58.8	1	10.4 (10.6)	93.7 (18.9)	17.2 (14.4)	19.7 (0.0)	27.2 (0.0)	22.9 (0.0)	20.41 (38.32)	7.92 (4.38)
SW ₇₀₀ ¹	50.6	100	67.2	0	5.6 (6.5)	98.6 (12.0)	9.9 (10.3)	nan (nan)	nan (nan)	nan (nan)	31.95 (64.36)	7.75 (4.45)
SW ₈₀₀ ¹	56.7	100	72.4	0	5.1 (6.6)	97.7 (11.0)	9.0 (10.4)	nan (nan)	nan (nan)	nan (nan)	44.53 (71.52)	7.7 (4.06)
SW ₈₀₀ ²	49.6	99.2	66.1	0	8.3 (9.4)	95.0 (15.9)	13.9 (13.2)	nan (nan)	nan (nan)	nan (nan)	30.52 (46.45)	7.82 (4.1)
SW ₉₀₀ ¹	64.3	100	78.3	0	4.2 (5.7)	94.7 (19.0)	7.5 (9.2)	nan (nan)	nan (nan)	nan (nan)	48.16 (80.31)	7.9 (4.35)
SW ₉₀₀ ³	42.2	92.4	58.0	0	15.0 (14.8)	81.5 (28.9)	22.7 (16.8)	nan (nan)	nan (nan)	nan (nan)	17.97 (29.56)	7.65 (4.67)
SW ₁₀₀₀ ¹	67.0	100	80.2	0	3.7 (4.7)	95.3 (18.3)	6.8 (7.9)	nan (nan)	nan (nan)	nan (nan)	57.83 (84.87)	7.91 (4.3)
SW ₁₀₀₀ ²	56.7	98.3	71.9	0	6.3 (6.9)	93.8 (19.1)	11.1 (10.9)	nan (nan)	nan (nan)	nan (nan)	47.26 (68.92)	7.98 (4.44)

Table 3: Individualization, segmentation and localization metrics for the best FCN-MN and SW detection models. Bolded cells denote the best among all the cells in the column corresponding to the same algorithm (i.e., the best among FCN-MN, and the best among SW). Underlined (bolded) cells denote the best overall FCN-MN and SW detection models.

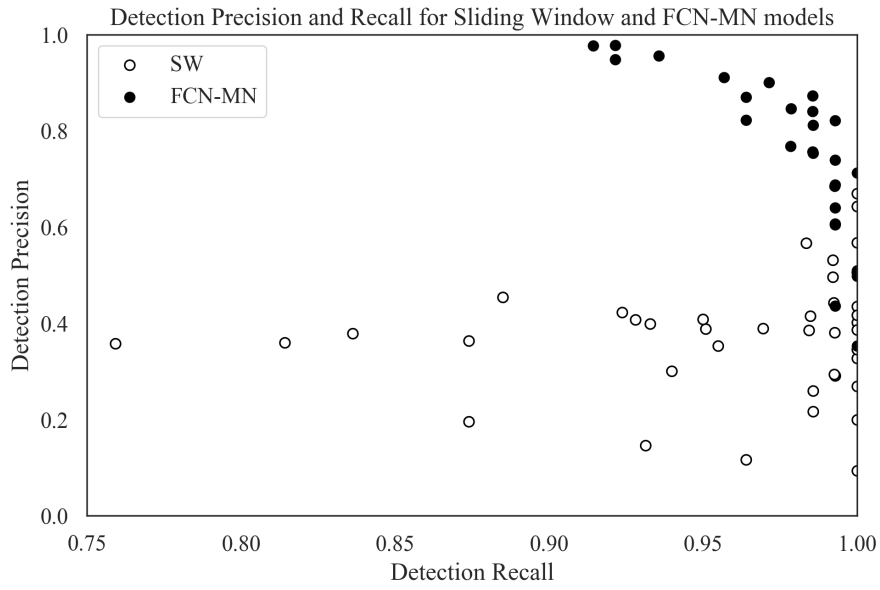


Figure 3: Precision-Recall scatterplots of the second and third columns of Table 3 discriminating the results for FCN-MN and SW with black and white dots, respectively. Each dot then represents the detection-precision and detection-recall computed over all images of the tests, for some particular configurations of hyperparameters. For FCN-MN, these would be the architecture, with values 8s, 16s and 32s, and threshold $\tau = \{0.1, 0.2, \dots, 0.9\}$, for a total of 27 black dots; while for SW these would be the patch sizes $\{100, 200, \dots, 1000\}$ and voting thresholds $\{1, 2, 3, 4\}$, for a total of a total of 40 white dots.

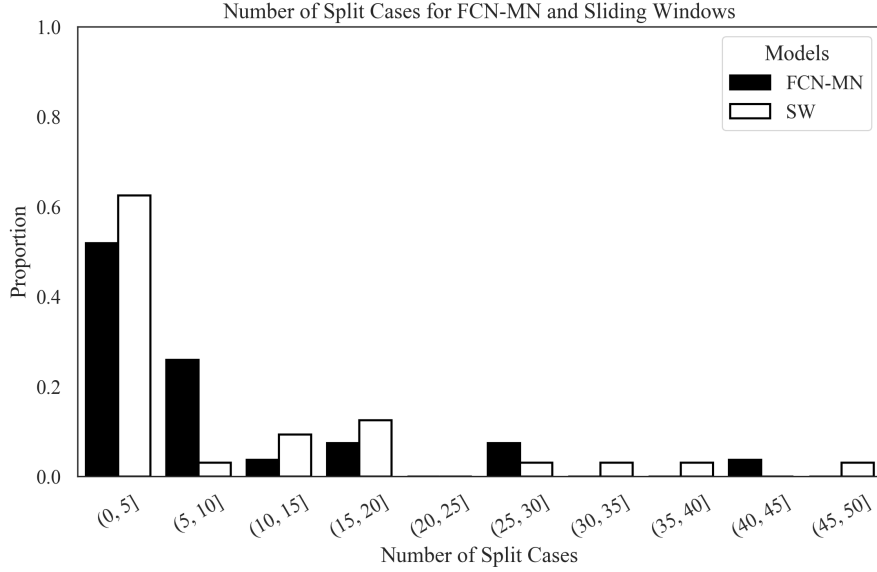


Figure 4: Histogram reporting the distribution of S for FCN-MN and SW in black and white bars, respectively. Each bar represents the proportion among all models (27 for FCN-MN and 40 for SW) that contains the number of splits indicated by the bin’s label. For instance, the first (from left to right) white bar indicates that almost 14% out of the 40 SW models contains between 0 to 5 splits.

SW models resulted in a total number splits of less than 5. Overall, the FCN-MN distribution is slightly more concentrated in the lower number of splits than the SW distribution, but in general both algorithms compare fairly, with no clear contender when compared on the average number of splits they produce.

3.2.2. Detailed analysis of segmentation metrics

As for the individualization metrics, we show in Figures 5a and 5b scatter plots for the segmentation precision and segmentation-recall, for the *correct detections* and *splits* cases, respectively. These correspond to their respective columns of (a non-summarized version of) Table 3, with the black and white dots representing the values of FCN-MN and SW detection models, respectively. The position of each dot in the plot corresponds to the mean segmentation-precision and mean segmentation-recall over all images in the test set, computed over the correctly detected components (splitted components, respectively) of the masks produced by the detection model associated to that dot. The standard deviation

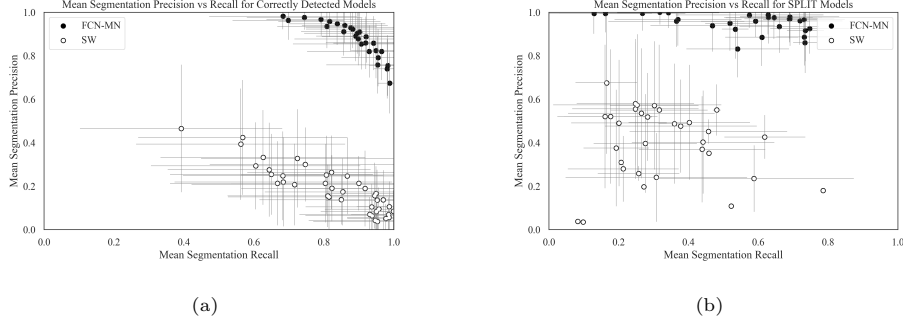


Figure 5: Segmentation Precision-Recall scatterplots reporting the results for FCN-MN and SW in black and white, respectively, with dots representing the average of segmentation precision and segmentation recall over all images in the test set (and bars representing standard deviations), with one dot per configuration of hyperparameters (27 for FCN-MN and 40 for SW). In (a), the averages were computed over the segmentation precision and recall of the correctly detected components, while in (b), the averages were computed over the segmentation precision and recall of the split components. Standar deviations.

of the recall (precision) is shown as a horizontal (vertical) bar. In Figure 5a (correctly detected), one can observe that all black dots (FCN-MN) are clustered on the upper-right corner of the graph, enclosed by a minimum precision of approximately 0.65 and minimum recall of approximately 0.60; while the white dots (SW) are clustered on the lower-right corner of the graph, with maximum precisions of 0.5 and recall ranging from approximately 0.35 to 1.0. Overall, both algorithms show relatively high recalls, but with FCN-MN reaching much larger precisions. We can point to the coarse detection of the SW method as the main cause for the low precision, as this is reduced when extra, false positives are present in the positive mask. In Figure 5b (splits), one can observe again the overwhelming improvements of FCN-MN over SW, with all (but one) SW cases presenting precisions under 60%, with the outlier showing a precision of nearly 70%, and a similar distribution of recall values.

We also report graphically the segmentation results for the false alarm, the *NA* for each of the 27 models of FCN and each of the 40 models of SW, i.e., for each cell in the one-before-last column of (a non-summarized version of) Table 3

Figure 6 shows these results grouped in the form of two histograms, one for the FCN-MN detection models (black) and one for the SW models (in white). Bars in the histogram represent the proportion of detection models whose mean

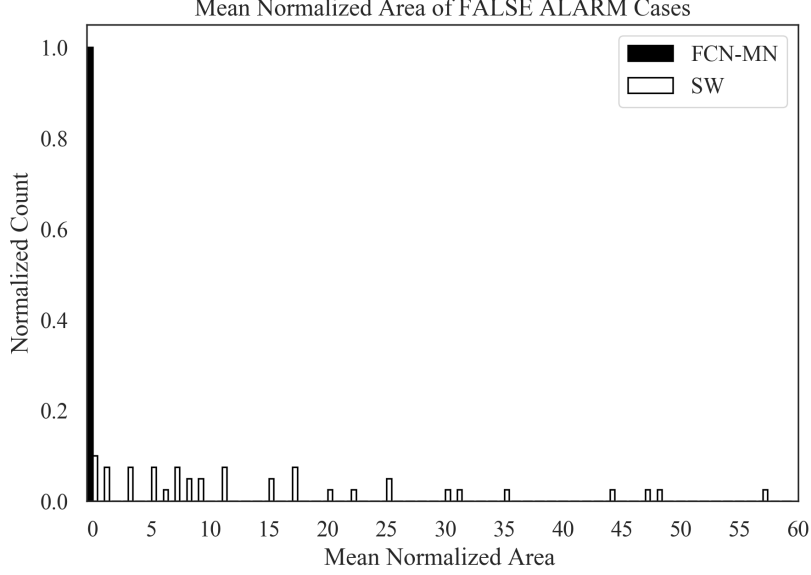


Figure 6: TODO:redactar

644 NA (over all all false alarm components of all images) falls within the interval of
 645 the bin. The more concentrated to the left, the better is the algorithm, as this
 646 indicates that more detection models for that algorithm resulted in smaller NA
 647 (on average). One can observe the histogram for FCN-MN considerably more
 648 concentrated at the left-most part of the histogram than that of SW, with all
 649 FCN-MN concentrated in a single bar at the left-most interval of $[0.0, 1.0)$. For
 650 SW the situation is rather different, with bars at intervals as far to the right as
 651 $[57.0, 58.0)$, that is, detection models with areas as large as 58 times the area of
 652 the bud.

653 3.2.3. Detailed analysis of localization metrics

654 To conclude, we present in this subsection a graphical representation of the
 655 localization results reported in Tab ??tab:TablaXX), that is, the *normalized*
 656 *distance*(ND) only for false alarms. This assumes that because they overlap
 657 the true bud, correctly detected and split cases should be close enough to the
 658 true bud to render it unnecessary any analysis on their distance. Instead, a false
 659 alarm can be arbitrarily far from the true bud.

660 Figure 7 summarizes the ND values reported in the corresponding column

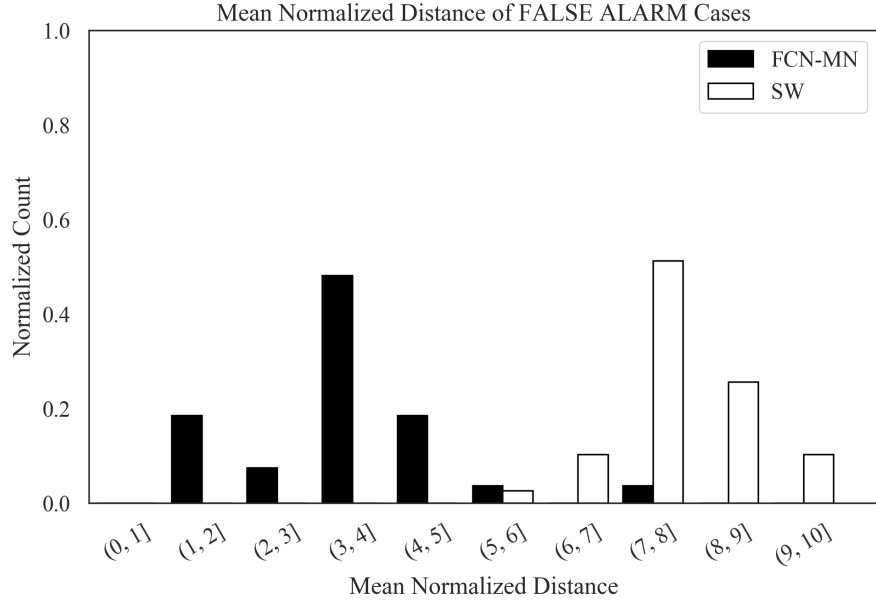


Figure 7: TODO:redactar

661 of the (non-summarized version) of Tab ??tab:TablaXX) in the form of two
 662 histograms, one for FCN-MN (black) and one for SW (white). Bars in the
 663 histogram represent the proportion of detection models (27 for FCN-MN and
 664 40 for SW) whose mean ND (over all all false alarm components of all images)
 665 falls within the interval of the bin. The more concentrated to the left, the better
 666 is the algorithm, as this indicates that more detection models for that algorithm
 667 resulted in smaller ND (on average).

668 Here again the advantage of FCN-MN over SW is clear, with the histogram
 669 for FCN-MN more concentrated to the left-most than that of SW, with the
 670 FCN-MN histogram running from the (0, 1] to the (7, 8] bin, whereas the SW
 671 histogram running from the (5, 6] towards the (9, 10] bin.

672 sectionDiscusión

673 En esta sección se discuten los resultados obtenidos por el enfoque propuesto
 674 en el contexto del problema de detección de yemas de vid, su impacto como
 675 herramienta para la medición de variables vitícolas de interés y los trabajos
 676 futuros.

677 This work introduces FCN-MN, a fully convolutional network with Mobile
 678 Net architecture ? for the detection of grapevine buds in 2D images captured

in natural field conditions, in winter (i.e., with no leaves nor bunches), and containing a maximum of one bud. The experimental results confirmed our main hypothesis, that the detection quality achieved by FCN-MN improves over the *scanning windows* detector (SW) ? in all three detection aspects: segmentation, individualization and localization. Being SW the best bud detector known to these authors, one can conclude that FCN-MN is a strong contender in the state-of-the-art for bud detectors.

But even improving over the state-of-the-art bud detectors one can still wonder if it can address the main *quality* and *throughput* requirements of a practical measurement of the bud related variables of Table 1..

Quality performance could be assessed by the metrics reported in Table 3, where in the best case FCN-MN shows a detection-precision and detection-recall of 97.7 and 100, respectively, a mean (and standard deviation) segmentation-precision and segmentation-recall for correctly detected of 98.1(0.6) and 98.8(3.4), respectively; and for splits 99.9(0.1) and 74.7(28.1), respectively. Also, for false alarms, a maximum *NA* of 0.04(0.09) a maximum *ND* of 0.04(0.22). However, these maximums correspond each to different FCN-MN detectors. A better assessment must be conducted for one single detector. For that, we picked FCN-MN_{16s}^{0.6} for showing balanced quality overall. This detector reaches detection precision and recall of 95.6 and 93.6 respectively, meaning than only 4.4% of all the detected connected components over all test images are false alarms, and that only 6.4% of all true buds could not be detected (i.e., resulted in detection failure). Also, $S = 3$, meaning only 3 of all detections were splitted, which on average has a segmentation precision of 99.4(0.6) and segmentation recall of 16.2(10.6). The recall is rather small, suggesting that the split is in fact the result of pixel wise detection of the bud so sparse that it got disconnected. In contrast, all remaining detections were correct (i.e., not splitted), reaching segmentation precisions of 92.2(8.7), a rather similar value to that of splits, but a much larger mean segmentation recall of 88.2(13.3). Overall, this resulted in a mean Dice measure for the correctly detected of 89.1(10.7); demonstrating a considerable (mean) coverage of the true bud, with only 11.8% of the buds pixels missing (on average), and only 7.8% of the detected pixels covering the background (on average). But more promising are the false alarm results, with

712 $NA = 0.08$ and $ND = 1.1$, showing that these components are rather small,
 713 covering only an area that is 8% in size of the total area of a bud (on average),
 714 and distant to the true bud by only 1.1(0.65) diameters.

715 Based on these results, ¿what quality one should expect when the FCN_{16s}^{0.6}
 716 detector takes part in the measurement of the bud related variables? For brevity
 717 we discuss this forthree variables from Table 1: *buds number*, *bud area*, and
 718 *length between nods*.

719 El caso del *buds number*, por ejemplo, requiere individualizar las yemas de
 720 la escena, por lo que su calidad se verá impactada sólo por la métricas de
 721 détection precision and recall (95.6 and 93.6 respectively). Para evaluar este
 722 impacto asumimos que una planta tiene aproximadamente en promedio 240
 723 yemas. El número de yemas por planta depende de muchos factores, como
 724 ser sistema de conducción, varietal, tipo de tratamiento, época del año, entre
 725 otros, por lo que este valor se define a modo indicativo para lograr un análisis
 726 aproximado. Para este caso, una detection precision de 95.6 resultaría en 11
 727 yemas contadas en exceso por planta; mientras que la recall de 93.6 resultaría
 728 en la omisión del conteo de 15 yemas. Además, este modelo produce 3 splits
 729 con dos componentes cada uno, i.e. un error de conteo por exceso del 3% yemas
 730 sobre las 140 yemas del testset. Particularmente en este análisis significa que
 731 se contarían 6 nuevas yemas de más, dando un total de 17 yemas en exceso,
 732 practicamente cancelandose con el error de omisión. Pero además, estos errores
 733 podrían en la práctica caracterizarse estadísticamente, permitiendo corregir las
 734 mediciones hacia valores más certeros.

735 La segunda variable de interés considerada es la *bud area*, donde, además
 736 de individualizar cada yema de la escena, es necesario segmentarla para estimar
 737 su área en píxeles. El análisis de individualización es análogo al del conteo de
 738 yemas, por lo que ahora se discuten sólo las métricas de segmentation. Del
 739 análisis desarrollado en los párrafos anteriores se puede concluir que los errores
 740 de segmentación por splits y false alarm tienen un bajo impacto en los resul-
 741 tados generales, y por ende en la estimación de *bud area*. Por otro lado, si se
 742 compensan los errores de segmentación para los correct detected (i.e. 11.8%
 743 of the buds pixels missing and 7.8% of the detected pixels covering the back-
 744 ground), el error de estimación del área es solo de un 4%. A efectos ilustrativos,

745 vemos que este error es menor al error de precisión resultante de medir el área
746 de una yema con un calibre. Si asumieramos que la forma de una yema se
747 ajusta a una circunferencia, y que el diámetro típico de una yema es de 5 mm
748 de diámetro, obtenemos un área de $19.63mm^2$. Siendo que un calibre tiene una
749 precisión es $0.1mm$, el error de precisión del área sería de $\pm 1.7mm^2$, equiva-
750 lente a un 8.6% del área total; un monto que duplica el error del 4% producido
751 por nuestro detector FCN-MN. A esta diferencia se le debe además sumar el
752 error de la medición manual resultante de asumir una forma circular de la yema,
753 aproximación innecesaria en el caso de FCN-MN.

754 Por último, consideremos el caso de la *distance between knots*, estimada por
755 la distancia entre yemas de una misma rama (por la cercanía entre yemas y
756 nudos), la cual involucra las operaciones de individualización y localización. De
757 nuevo, el análisis de individualización es análogo al del conteo de yemas, que en
758 este caso resultará en el reporte de más de una distancia debido a la detección
759 de más de una componente por yema. Entre estas distancias, entendemos que
760 el peor caso puede darse entre los false alarms, siendo estos los más alejados de
761 la true bud, y entre dos yemas ocurre cuando las false alarms están a distancia
762 ND del lado más alejado de la otra yema. En promedio, $ND = 1.1$, que de
763 acuerdo al diámetro típico de las yemas de vid equivale a aprox. 5mm, un valor
764 muy menor a las distancias típicas de yemas de aproximadamente 30cm, i.e.,
765 alrededor de un 3.3% de error en la estimación de la distancia entre buds/knots.

766 Vemos que los errores de mayor impacto ocurren por el exceso u omisión de
767 connected components, con el error de exceso exacerbado por el hecho de asociar
768 buds detectadas con connected components individuales. Una mejora posible
769 para mitigar estos errores consistiría en aplicar algunos post-procesamientos.
770 Uno de ellos es el *spatial clustering* de los connected components que los agrupe
771 por cercanía. One could expect this to improve the results based on the
772 small areas of split and false alarm components. On one hand, due to the
773 closeness to the true bud of the false alarms (small ND), as well as the splits
774 and correctly detected components (they overlap with it); and the fact that
775 true buds in real plants are typically tens or even hundreds of bud diameters
776 apart, a simple spatial clustering of the components would connect all these
777 components together as one single, and correct, bud detection. Second, due to

778 their small area, if clustered together, the false alarm components would only
779 slightly reduce the segmentation precision. Otro posible post-procesamiento
780 consistiría en descartar connected components pequeños, por ejemplo, cuya area
781 en pixeles normalizada respecto al area total detectada (suma de las areas de
782 todos los connected components) sea menor a cierto umbral. Podrían esperarse
783 mejoras con este post-procesamiento dado que los resultados en este trabajo
784 muestran que los false alarms presentan areas pequeñas en relacion al true bud.
785 Por último, podrían considerarse filtros de connected components basados en
786 la estructura de la planta, por ejemplo, descartando connected components que
787 estan lejos (o no presentan overlap) con las ramas.

788 Also, one could consider in future works some improvements that overcome
789 the limitations for a practical use mentioned above: (i) no associations between
790 parts of plants of different images, (ii) distance and area measurements are in
791 pixels, (iii) only 3D geometry, (iv) lack of knowledge of the underlying plant
792 structure, and (v) need of images with no leaves.

793 One could consider extending to buds the work of [Santos et al. \(2020\)](#) that
794 addresses limitation (i) for grape bunches. Limitation (ii) could be easily ad-
795 dressed by adding to the visual scene some marker with known dimensions. This,
796 however, requires such a marker in every image captured, a problem that could
797 be overcome by first producing a calibrated 3D reconstruction of the scene,
798 i.e., a 3D reconstruction calibrated with a single marker in one of its frames ?.
799 This way, every 2D image could be calibrated against the 3D model, omitting
800 the need of a marker. In addition, a 3D reconstruction of the scene could ad-
801 dress limitation (iii) by locating the detected buds in 3D space, following, for
802 instance, the approach taken by [Díaz et al. \(2018\)](#).

803 Finally, a solution to limitations (iv) and (v) would require an integrated
804 solution involving the detection in 3D of branches and leaves, respectively.

805 4. Conclusions

806 TODO: Conclusions

807 **Acknowledgments**

808 This work was funded by the National Technological University (UTN), the
809 National Council of Scientific and Technical Research (CONICET), Argentina,
810 and the National Fund for Scientific and Technological Promotion (FONCyT),
811 Argentina. We thank the National Agricultural Technology Institute (INTA)
812 for offering their vineyards to capture the images used in this work.

813 **References**

814 **References**

- 815 Berenstein, R., Shahar, O.B., Shapiro, A., Edan, Y., 2010. Grape clusters
816 and foliage detection algorithms for autonomous selective vineyard sprayer.
817 *Intelligent Service Robotics* 3, 233–243.
- 818 Borgogno-Mondino, E., Lessio, A., Tarricone, L., Novello, V., de Palma, L.,
819 2018. A comparison between multispectral aerial and satellite imagery in
820 precision viticulture. *Precision Agriculture* 19, 195–217.
- 821 Bramley, R.G., 2009. Lessons from nearly 20 years of precision agriculture
822 research, development, and adoption as a guide to its appropriate application.
823 *Crop and Pasture Science* 60, 197–217.
- 824 Chum, O., Zisserman, A., 2007. An exemplar model for learning object classes,
825 in: *2007 IEEE Conference on Computer Vision and Pattern Recognition*,
826 IEEE. pp. 1–8.
- 827 Collins, C., Wang, X., Lesefko, S., De Bei, R., Fuentes, S., 2020. Effects of
828 canopy management practices on grapevine bud fruitfulness. *OENO One* 54,
829 313–325.
- 830 Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C., 2004. Visual cat-
831 egorization with bags of keypoints, in: *Workshop on statistical learning in*
832 *computer vision, ECCV, Prague*. pp. 1–2.
- 833 Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detec-
834 tion, in: *2005 IEEE Computer Society Conference on Computer Vision and*
835 *Pattern Recognition (CVPR’05)*, pp. 886–893 vol. 1.

836 Diago, M.P., Correa, C., Millán, B., Barreiro, P., Valero, C., Tardaguila, J.,
837 2012. Grapevine yield and leaf area estimation using supervised classification
838 methodology on rgb images taken under field conditions. *Sensors* 12, 16988–
839 17006.

840 Díaz, C.A., Pérez, D.S., Miatello, H., Bromberg, F., 2018. Grapevine buds
841 detection and localization in 3d space based on structure from motion and 2d
842 image classification. *Computers in Industry* 99, 303–312.

843 Dice, L.R., 1945. Measures of the amount of ecologic association between species.
844 *Ecology* 26, 297–302.

845 Divvala, S.K., Hoiem, D., Hays, J.H., Efros, A.A., Hebert, M., 2009. An em-
846 pirical study of context in object detection, in: 2009 IEEE Conference on
847 computer vision and Pattern Recognition, IEEE. pp. 1271–1278.

848 Ferrari, V., Fevrier, L., Jurie, F., Schmid, C., 2007. Groups of adjacent contour
849 segments for object detection. *IEEE transactions on pattern analysis and*
850 *machine intelligence* 30, 36–51.

851 Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-
852 Gonzalez, P., Garcia-Rodriguez, J., 2018. A survey on deep learning tech-
853 niques for image and video semantic segmentation. *Applied Soft Computing*
854 70, 41–65.

855 Grimm, J., Herzog, K., Rist, F., Kicherer, A., Töpfer, R., Steinhage, V., 2019.
856 An adaptable approach to automated visual detection of plant organs with
857 applications in grapevine breeding. *Biosystems Engineering* 183, 170–183.

858 Han, D., 2013. Comparison of commonly used image interpolation methods,
859 in: *Proceedings of the 2nd international conference on computer science and*
860 *electronics engineering*, Atlantis Press.

861 Herzog, K., Kicherer, A., Töpfer, R., 2014a. Objective phenotyping the time of
862 bud burst by analyzing grapevine field images, in: *XI International Confer-*
863 *ence on Grapevine Breeding and Genetics* 1082, pp. 379–385.

864 Herzog, K., et al., 2014b. Initial steps for high-throughput phenotyping in
865 vineyards. *Australian and New Zealand Grapegrower and Winemaker* , 54.

866 Hirano, Y., Garcia, C., Sukthankar, R., Hoogs, A., 2006. Industry and ob-
867 ject recognition: Applications, applied research and challenges, in: Toward
868 category-level object recognition. Springer, pp. 49–64.

869 Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T.,
870 Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural
871 networks for mobile vision applications. arXiv preprint arXiv:1704.04861 .

872 Intrigliolo, D., Castel, J., 2007. Evaluation of grapevine water status from trunk
873 diameter variations. *Irrigation Science* 26, 49–59.

874 Jaccard, P., 1912. The distribution of the flora in the alpine zone. 1. *New*
875 *phytologist* 11, 37–50.

876 Kahng, M., Andrews, P.Y., Kalro, A., Chau, D.H.P., 2017. A cti v is: Visual
877 exploration of industry-scale deep neural network models. *IEEE transactions*
878 *on visualization and computer graphics* 24, 88–97.

879 Kaymak, Ç., Uçar, A., 2019. A brief survey and an application of semantic
880 image segmentation for autonomous driving, in: *Handbook of Deep Learning*
881 *Applications*. Springer, pp. 161–200.

882 Kliewer, W.M., Dokoozlian, N.K., 2005. Leaf area/crop weight ratios of
883 grapevines: influence on fruit composition and wine quality. *American Jour-*
884 *nal of Enology and Viticulture* 56, 170–181.

885 Kornblith, S., Shlens, J., Le, Q.V., 2019. Do better imagenet models trans-
886 fer better?, in: *Proceedings of the IEEE conference on computer vision and*
887 *pattern recognition*, pp. 2661–2671.

888 Lampert, C.H., Blaschko, M.B., Hofmann, T., 2008. Beyond sliding windows:
889 Object localization by efficient subwindow search, in: *2008 IEEE conference*
890 *on computer vision and pattern recognition*, IEEE. pp. 1–8.

891 Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian,
892 M., Van Der Laak, J.A., Van Ginneken, B., Sánchez, C.I., 2017. A survey on
893 deep learning in medical image analysis. *Medical image analysis* 42, 60–88.

- 894 Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for
895 semantic segmentation, in: Proceedings of the IEEE conference on computer
896 vision and pattern recognition, pp. 3431–3440.
- 897 Lorenz, D., Eichhorn, K., Bleiholder, H., Klose, R., Meier, U., Weber, E., 1995.
898 Growth stages of the grapevine: Phenological growth stages of the grapevine
899 (*vitis vinifera* l. ssp. *vinifera*)—codes and descriptions according to the ex-
900 tended bbch scale. Australian Journal of Grape and Wine Research 1, 100–
901 103.
- 902 Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints.
903 International journal of computer vision 60, 91–110.
- 904 Matese, A., Di Gennaro, S.F., 2015. Technology in precision viticulture: A state
905 of the art review. International journal of wine research 7, 69–81.
- 906 May, P., 2000. From bud to berry, with special reference to inflorescence and
907 bunch morphology in *vitis vinifera* l. Australian Journal of Grape and Wine
908 Research 6, 82–98.
- 909 Ning, C., Zhou, H., Song, Y., Tang, J., 2017. Inception single shot multibox
910 detector for object detection, in: 2017 IEEE International Conference on
911 Multimedia & Expo Workshops (ICMEW), IEEE. pp. 549–554.
- 912 Noyce, P.W., Steel, C.C., Harper, J.D., Wood, R.M., 2016. The basis of defolia-
913 tion effects on reproductive parameters in *vitis vinifera* l. cv. chardonnay lies
914 in the latent bud. American Journal of Enology and Viticulture 67, 199–205.
- 915 Nuske, S., Achar, S., Bates, T., Narasimhan, S., Singh, S., 2011. Yield estima-
916 tion in vineyards by visual grape detection, in: 2011 IEEE/RSJ International
917 Conference on Intelligent Robots and Systems, IEEE. pp. 2352–2358.
- 918 Oguz, I., Carass, A., Pham, D.L., Roy, S., Subbana, N., Calabresi, P.A., Yushke-
919 vich, P.A., Shinohara, R.T., Prince, J.L., 2017. Dice overlap measures for
920 objects of unknown number: application to lesion segmentation, in: Interna-
921 tional MICCAI Brainlesion Workshop, Springer. pp. 3–14.
- 922 Ozdemir, G., Sessiz, A., Pekitkan, F.G., 2017. Precision viticulture tools to
923 production of high quality grapes. Sci. Pap. Ser. B Hortic 61, 209–218.

924 Pan, S.J., Yang, Q., 2009. A survey on transfer learning. *IEEE Transactions on*
925 *knowledge and data engineering* 22, 1345–1359.

926 Pellegrino, A., Lebon, E., Simonneau, T., Wery, J., 2005. Towards a simple
927 indicator of water stress in grapevine (*vitis vinifera* l.) based on the differential
928 sensitivities of vegetative growth components. *Australian Journal of Grape*
929 *and Wine Research* 11, 306–315.

930 Pérez, D.S., Bromberg, F., Diaz, C.A., 2017. Image classification for detection
931 of winter grapevine buds in natural conditions using scale-invariant features
932 transform, bag of features and support vector machines. *Computers and*
933 *electronics in agriculture* 135, 81–95.

934 Poni, S., Gatti, M., Palliotti, A., Dai, Z., Duchêne, E., Truong, T.T., Ferrara,
935 G., Matarrese, A.M.S., Gallotta, A., Bellincontro, A., et al., 2018. Grapevine
936 quality: A multiple choice issue. *Scientia horticulturae* 234, 445–462.

937 Reynolds, A.G., Heuvel, J.E.V., 2009. Influence of grapevine training systems
938 on vine growth and fruit composition: a review. *American Journal of Enology*
939 *and Viticulture* 60, 251–268.

940 Rowley, H.A., Baluja, S., Kanade, T., 1996. Human face detection in visual
941 scenes, in: *Advances in Neural Information Processing Systems*, pp. 875–881.

942 Rudolph, R., Herzog, K., Töpfer, R., Steinhage, V., 2018. Efficient identi-
943 fication, localization and quantification of grapevine inflorescences in un-
944 prepared field images using fully convolutional networks. *arXiv preprint*
945 *arXiv:1807.03770* .

946 Sánchez, L.A., Dokoozlian, N.K., 2005. Bud microclimate and fruitfulness in
947 *vitis vinifera* l. *American Journal of Enology and Viticulture* 56, 319–329.

948 Santos, T.T., de Souza, L.L., dos Santos, A.A., Avila, S., 2020. Grape detection,
949 segmentation, and tracking using deep neural networks and three-dimensional
950 association. *Computers and Electronics in Agriculture* 170, 105247.

951 Seng, K.P., Ang, L.M., Schmidtke, L.M., Rogiers, S.Y., 2018. Computer vision
952 and machine learning for viticulture technology. *IEEE Access* 6, 67494–67510.

953 Shelhamer, E., Long, J., Darrell, T., 2017. Fully convolutional networks for
954 semantic segmentation. *IEEE transactions on pattern analysis and machine*
955 *intelligence* 39, 640–651.

956 Shorten, C., Khoshgoftaar, T.M., 2019. A survey on image data augmentation
957 for deep learning. *Journal of Big Data* 6, 60.

958 Siam, M., Gamal, M., Abdel-Razek, M., Yogamani, S., Jagersand, M., 2018.
959 Rtseg: Real-time semantic segmentation comparative study, in: 2018 25th
960 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 1603–
961 1607.

962 Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-
963 scale image recognition. *CoRR* abs/1409.1556.

964 Tardaguila, J., Diago, M., Blasco, J., Millán, B., Cubero, S., García-Navarrete,
965 O., Aleixos, N., 2012. Automatic estimation of the size and weight of
966 grapevine berries by image analysis, in: *Proc. CIGR AgEng*.

967 Tardáguila, J., Diago, M.P., Millan, B., Blasco, J., Cubero, S., Aleixos, N., 2012.
968 Applications of computer vision techniques in viticulture to assess canopy
969 features, cluster morphology and berry size, in: *I International Workshop on*
970 *Vineyard Mechanization and Grape and Wine Quality* 978, pp. 77–84.

971 Tarry, C., Wspanialy, P., Veres, M., Moussa, M., 2014. An integrated bud
972 detection and localization system for application in greenhouse automation,
973 in: 2014 Canadian Conference on Computer and Robot Vision, IEEE. pp.
974 344–348.

975 Taylor, J.A., Dresser, J., Hickey, C.C., Nuske, S., Bates, T.R., 2019. Consid-
976 erations on spatial crop load mapping. *Australian journal of grape and wine*
977 *research* 25, 144–155.

978 Tilgner, S., Wagner, D., Kalischewski, K., Velten, J., Kummert, A., 2019. Multi-
979 view fusion neural network with application in the manufacturing industry,
980 in: 2019 IEEE International Symposium on Circuits and Systems (ISCAS),
981 IEEE. pp. 1–5.

- 982 Vapnik, V., 2013. The nature of statistical learning theory. Springer science &
983 business media.
- 984 Wang, X., Han, T.X., Yan, S., 2009. An hog-lbp human detector with partial
985 occlusion handling, in: 2009 IEEE 12th international conference on computer
986 vision, IEEE. pp. 32–39.
- 987 Whalley, J., Shanmuganathan, S., 2013. Applications of image processing in
988 viticulture: A review .
- 989 Whelan, B., McBratney, A., Viscarra Rossel, R., 1996. Spatial prediction for
990 precision agriculture, in: Proceedings of the Third International Conference
991 on Precision Agriculture, Wiley Online Library. pp. 331–342.
- 992 Xu, S., Xun, Y., Jia, T., Yang, Q., 2014. Detection method for the buds
993 on winter vines based on computer vision, in: 2014 Seventh International
994 Symposium on Computational Intelligence and Design, IEEE. pp. 44–48.
- 995 Zhao, F., Rong, D., Liping, L., Chenlong, L., 2018. Research on stalk crops
996 internodes and buds identification based on computer vision. MS&E 439,
997 032080.