

Deep Learning for 2D grapevine bud detection

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

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

^b*Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET).*

Abstract

Visual inspection is a task necessary to measure relevant variables in viticulture and is susceptible to being automated with computer vision methods. Bud detection is central for various of these tasks such as: measurement of buds' sunlight exposure, autonomous pruning, bud counting, type-of-bud classification, bud geometric characterization, internode length, and bud development stage, among others. This paper presents a method for grapevine bud detection based on a *Fully Convolutional Networks Mobile-Net* architecture. To validate its performance we compare it on the detection task with the known state-of-the-art method for bud detection, showing improvements over three of the aspects of detection: *segmentation*, *correspondence identification* and *localization*. In its best version of configuration parameters, our approach showed a detection precision of 95.6%, detection recall of 93.6%, and a mean Dice coefficient of 89.1% for correct detection segmentations. The paper concludes with a discussion on the advantages of our approach for real-world applications.

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

*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 Diaz), `fbromberg@frm.utn.edu.ar` (Facundo Bromberg)

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 years viticulturists have been producing models of the most relevant plant processes (i.e. fruit quality and yield, soil profiling, vine health), and they have been recollecting a diverse collection 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 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. The list is long, with examples of these variables being berry maturity, number, weight, size and volume; cluster compactness, morphology such as length, width, size, and elongation, as well as cluster volume, number and weight; buds burst, number and size; flowers number; leaf area; shoot length; pruning weight; canopy density; among others (Institute, a,b)), 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 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). 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 area, 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) indicate what specific aspects of the detection are required for that variable: (i) whether it requires a good *segmentation*, i.e., the discrimination of which pixels in the scene

Variable	(i)	(ii)	(iii)	
Buds number		x		none
Bud area	x	x		none
Type-of-bud classification	x	x		plant structure (trunk and canes)
Bud development stage	x	x		classifier over bud mask
Internode length (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: A non-exhaustive list of important bud related variables, 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) indicate what of the three aspects of the detection it requires: segmentation, correspondence identification, or localization, respectively.

67 correspond to buds and which ones correspond to the background (no-bud); ii)
68 a good *correspondence identification*, i.e., discrimination of bud pixels as be-
69 longing to different buds; or (iii) a good *localization*, i.e., the localization of the
70 bud within the scene; respectively. For instance, tomemos por caso la variable
71 *buds number*. De ser posible identificar correctamente las correspondencias, the
72 buds number coincide directamente con el conteo de detecciones. Por el con-
73 trario, para *type-of-bud classification*, además de identificar correspondencias,
74 la segmentación de la parte de la imagen perteneciente a la yema es necesaria
75 para poder así alimentar a un clasificador con la información visual relevante,
76 minimizando el ruido producto de pixeles del background. Por último, para
77 medir la *incidence of sunlight on the bud*, no es necesaria la segmentación, sino
78 tan solo una buena localización de la yema, además de la leaves 3D superficial
79 geometry.

80 A good detector, therefore, should be evaluated on all three aspects of seg-
81 mentation, correspondence identification and localization. This is easy for our
82 detector as its implementation first produces a segmentation mask, which is
83 then post-processed to produce the correspondence identification and localiza-
84 tion. Los detalles de este enfoque se detallan en la Sección~2. El análisis de
85 los resultados de detección presentado en la Sección~3 muestra que este en-
86 foque resulta superador a los algoritmos del estado del arte para la detección

de yemas de vid. Finalmente en la Sección~4 se discuten el alcance, las limitaciones de los resultados obtenidos para la detección de yemas, la suficiencia de la performance alcanzada para la medición de una selección de las variables de la Tabla~3, como también se destacan las conclusiones más importantes, los futuros trabajos y posibles mejoras.

1.1. Related work

En la literatura se pueden encontrar una gran variedad de trabajos que emplean algoritmos de computer vision y machine learning para adquirir información sobre los viñedos (Seng et al., 2018), como ser berry and bunch detection (Nuske et al., 2011), fruit size and weight estimation (Tardaguila et al., 2012), leaf area indices and yield estimation (Diago et al., 2012), plant phenotyping (Herzog et al., 2014a,b), autonomous selective spraying (Berenstein et al., 2010), y más (Tardaguila et al., 2012; Whalley and Shanmuganathan, 2013). Entre los algoritmos de computer que se destacan en los últimos años, the *artificial neural networks* han despertado gran interés en la industria para llevar a cabo diversas tareas de reconocimiento visual (Hirano et al., 2006; Kahng et al., 2017; Tilgner et al., 2019). Particularmente las *Convolutional Neural Networks* (CNNs) se han convertido en el enfoque dominante de machine learning para el reconocimiento visual de objetos (Ning et al., 2017). Dos estudios recientes han aplicado exitosamente técnicas de reconocimiento visual basado en *deep learning networks* para identificar variables vitícolas que permitan estimar la producción en viñedos. Uno de ellos Grimm et al. (2019) utiliza una FCN para realizar segmentación de órganos de la planta de vid como los young shoots, pedicels, flower, buds or grapes. El segundo Rudolph et al. (2018) utiliza imágenes de vid en condiciones de campo que son segmentadas utilizando una CNN para detectar inflorescences y sobre esas regiones segmentadas se aplica el algoritmo circle Hough Transform para detectar las flowers buds.

Varios trabajos apuntan tanto a detectar como a localizar buds en diferentes tipos de cultivos mediante sistemas de reconocimiento visual autónomo. For instance Tarry et al. (2014) presents an integrated system for chrysanthemum bud detection that can be used to automate labour intensive tasks in floriculture greenhouses. More recently Zhao et al. (2018) presents a system of computer vision that is used to identify the internodes and buds of stalk crops. Según

nuestro conocimiento y el mejor de nuestros esfuerzos de búsqueda, existen al menos cuatro trabajos que abordan el problema de la detección de yemas específicamente de la vid mediante sistemas de reconocimiento visual autónomo. Los trabajos presentados por Xu et al. (2014), Herzog et al. (2014b) y Pérez et al. (2017) aplican diferentes técnicas para realizar detección 2D en imágenes que involucra diferentes algoritmos de computer y machine learning. Además, Díaz et al. (2018) introduce un workflow para localizar yemas en el espacio 3D. A continuación se presentan los detalles más relevante de cada uno.

El trabajo de Xu et al. (2014), presenta un algoritmo de detección de yemas utilizando imágenes RGB capturadas indoor y condiciones controladas de iluminación y fondo. Específicamente para establecer un groundwork para un sistema de podado autónomo en invierno. Los autores aplican un filtro por umbral para discriminar el fondo del esqueleto de la planta, resultando en una imagen binaria. Asumen que la forma de las yemas son similares a esquinas y aplican el algoritmo *Harris corner detector* sobre la imagen binaria para detectarlas. Este proceso obtiene un recall de 0.702, es decir el 70.2% de la yemas fueron detectadas.

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

En Pérez et al. (2017), presenta un enfoque para la clasificación de imágenes de yemas en invierno, mediante un enfoque que emplea *SVM* como clasificador y *Bag of Features* para computar descriptores visuales. Reportan un recall superior a 90% y una precisión de 86% cuando se clasifican imágenes que contienen al menos el 60% de una yema y una proporción del 20-80% de píxeles yema vs píxeles no-yema. Argumentan que este clasificador puede ser utilizados en algoritmos para localización 2D del tipo *sliding windows* debido a la robustez

153 ante la variación en tamaño y posición de la ventana. Es esta idea justamente
154 la que se ha reproducido en el presente trabajo para implementar el enfoque de
155 línea base basado en sliding windows y clasificador de patches.

156 Finalmente, en [Díaz et al. \(2018\)](#) se introduce un workflow para localización
157 de yemas en el espacio 3D. El workflow consta de 5 etapas. La primera real-
158 iza una reconstrucción a partir de varias imágenes RGB de una nube 3D de
159 puntos correspondientes a la estructura de la planta de vid. La segunda etapa
160 aplica un metodo de deteccion 2D utilizando una técnica de sliding window y
161 clasificación de patches. La etapa siguiente utiliza un esquema de votos para
162 clasificar cada punto de la nube como yema o no yema. La cuarta etapa aplica
163 el algoritmo de clustering *DBSCAN* para agrupar puntos de la nube que corre-
164 sponden a una yema. Finalmente en la quinta etapa se realiza la localización,
165 obteniendo las coordenadas del centro de masa de cada cluster de puntos 3D.
166 Reportan un recall de 45% con una precision de 100% y un error de localización
167 de aproximadamente 1.5cm, ó 3 diámetros de yema.

168 Si bien estos trabajos representan un gran avance en relación a la prob-
169 lemática de detección y localización de yemas, todavía sufren al menos una de
170 las siguientes limitaciones: (i) uso de fondo artificial en exteriores; (ii) ilumi-
171 nación controlada en interiores; (iii) necesidad de interacción con el usuario; (iv)
172 detección de yemas en etapas de desarrollo muy avanzado; (v) bajo recall de
173 detección/clasificación de yemas, y (vi) aunque algunos de estos trabajos real-
174 izan algún proceso de segmentación como parte del enfoque, ninguno apunta a
175 segmentar la yema, o reportar métricas de la calidad de la segmentación real-
176 izada. Estas limitaciones representan una importante barrera para el desarrollo
177 efectivo de herramientas de medición de variables asociadas a las yemas.

178 2. Materials and Methods

179 In this section we describe the main contribution of this work, the deep
180 learning setup for the detection of grapevine buds in 2D images of vine plants
181 captured in natural conditions. Subsection~2.1 provides the details on the *en-*
182 *coder-decoder* transfer learning architecture and the pre-training chosen for its
183 encoder. Los resultados de detección alcanzados por este enfoque son contrasta-
184 dos con un método de detección de yemas descrito en [Pérez et al. \(2017\)](#). En

este trabajo los autores presentan un clasificador de imágenes entre aquellas que contienen yemas y las que no, y sugieren su uso para la detección de yemas basado en la idea de *sliding windows*, que dada una imagen de una escena vitícola la subdivide en un conjunto de *patches* o regiones más pequeñas (Pérez et al., 2017), para luego detectar yemas determinando si cierto patch contiene o no una yema usando el clasificador de imágenes. En la subsection~2.2 se describe nuestra implementación de un detector basado en este diseño.

We then conclude the section with subsection~2.3 that provides details on the training of both methods including details on the collection of images used for its training.

2.1. Fully Convolutional Network with MobileNet (FCN-MN)

Como se describió en la introducción, el enfoque propone el uso de algoritmos de visión computacional para: (i) *segmentar* las yemas *clasificando* cuales píxeles de la escena corresponden a yema y cuales píxeles corresponden al background (no-yema), (ii) *identificar correspondencias* para las yemas distinguiendo entre aquellos pixeles que pertenecen a diferentes yemas en la escena observada, y (iii) *localizar* cada yema en la escena. Para la operación de segmentación, i.e., clasificación de pixeles, se toma como base la fully convolutional network introducida en (Long et al., 2015), y se entrena para el problema específico de segmentación de yemas de vid. El siguiente apartado 2.1.1 describe en detalle la arquitectura considerada para estas redes. La fully convolutional network resultante devuelve un mapa de probabilidad de igual escala que la imagen original, donde el valor de un píxel representa la probabilidad de que el píxel correspondiente en la imagen de entrada pertenezca a una yema. Para obtener una máscara binaria se aplica a cada píxel un umbral de clasificación τ , clasificando al pixel como yema (no-yema) si su probabilidad es mayor (menor) a τ . Para identificar correspondencias de las yemas se toma esta máscara binaria y se realiza un post-procesamiento para determinar que dos píxeles yema corresponden a una misma yema siempre y cuando pertenezcan a un mismo componente conectado, i.e., si los une alguna secuencia de píxeles yema contiguos. Finalmente, para la localización de objetos existen diversas alternativas entre las que encuentran *bounding box*, *pixel-wise segmentation*, *contorno* y *centro de masa del objeto*

(Lampert et al., 2008). En este trabajo se tomó la última, eligiendo localizar a las yemas por el *centro de masa* de su componente conectado.

2.1.1. Encoder-decoder architecture

Para el clasificador de píxeles se consideraron las tres versiones 32s, 16s y 8s de las *fully convolutional network* introducidas originalmente por Long et al. (2015), por haber sido utilizadas con excelentes resultados en muchas aplicaciones de segmentación de imágenes Litjens et al. (2017); Garcia-Garcia et al. (2018); Kaymak and Uçar (2019). Estas redes presentan arquitecturas características con dos partes bien distinguibles: *encoder* y *decoder* (ver~1). El encoder consiste en una convolutional neural network que realiza un *downsampling* de una imagen de entrada en un conjunto de features mediante operaciones de convolución, para producir un conjunto de *feature maps*, i.e. una representación abstracta de la imagen que captura información semántica y contextual, pero que descarta información espacial de grano fino. Estas operaciones reducen las dimensiones espaciales de la imagen a medida que se avanza más profundo en la red, resultando en feature maps de tamaño $1/n$ del tamaño de la imagen de entrada, donde n es el factor de downsampling. El decoder es una sub-red de *upsampling*, que toma el conjunto de feature maps de baja resolución y los proyecta al espacio de píxeles, aumentando la resolución para producir una máscara de segmentación (o clasificación densa de píxeles) con las mismas dimensiones de la imagen de entrada. Esta operación se implementa como una red de transposed convolutions con parámetros entrenables, también conocidas como upsample convolutions Shelhamer et al. (2017).

Por otra parte, para refinar la calidad de la segmentación, se suelen utilizar conexiones que sobrepasan al menos una capa de la red, llamadas *skip connections*. Éstas se utilizan para transferir información espacial local desde las capas internas del encoder directamente al decoder. En general, estas conexiones mejoran los resultados de segmentación, ya que mitigan la pérdida de información espacial permitiendo al decoder incorporar información de feature maps internos, aunque su impacto puede variar según la skip architecture que se proponga. En Long et al. (2015) se proponen tres skip architectures: la 32s sin información de capas internas del encoder; la 16s que suma información espacial de capas profundas del encoder; y la 8s, que suma información espacial de capas profun-

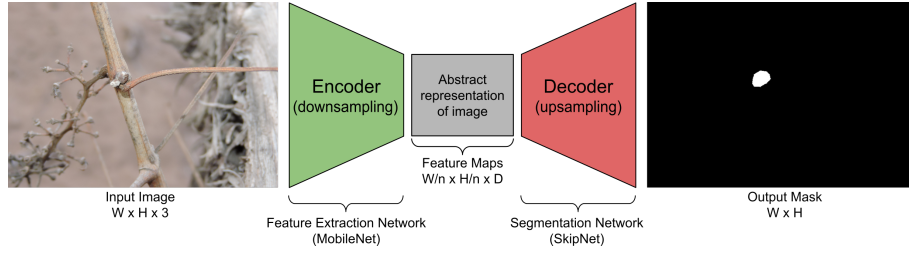


Figure 1: Esquema de la arquitectura de red FCN-MN propuesta en este trabajo, basada en la FCN propuesta por [Shelhamer et al. \(2017\)](#), reemplazando su encoder de extracción de features por las redes MobileNet [Howard et al. \(2017\)](#), lo que produce features maps con un factor de downsampling 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.

das y menos profundas del encoder. Los detalles de estas arquitecturas quedan fuera del alcance de este trabajo, pero pueden consultarse en [Long et al. \(2015\)](#) y [Shelhamer et al. \(2017\)](#). Dado que los resultados reportados en la literatura no son concluyentes respecto a que arquitectura es mejor [Long et al. \(2015\)](#); [Shelhamer et al. \(2017\)](#), en este trabajo se consideran las tres alternativas.

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

270 2.2. Sliding Windows detector

271 En esta sección se describe el enfoque propuesto por Pérez et al. (2017)
272 para clasificación de imágenes de yema y una implementación del mismo para
273 detección basada en sliding windows descrita en el trabajo original. A este
274 enfoque de detección no referimos como SW de aquí a lo que resto del paper.

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

281 Las técnicas sliding windows comprenden una familia de algoritmos amplia-
282 mente utilizados en el pasado como parte de diversos enfoques para localización
283 de objetos con bounding boxes (Divvala et al., 2009; Wang et al., 2009; Chum
284 and Zisserman, 2007; Ferrari et al., 2007; Dalal and Triggs, 2005; Rowley et al.,
285 1996). En estos algoritmos, cada imagen es escaneada densamente desde un ex-
286 tremo de la imagen (e.g. esquina superior izquierda) hasta el otro extremo (e.g.
287 esquina inferior derecha) mediante una ventana deslizante rectangular en difer-
288 entes escalas y diferentes desplazamientos, extrayendo sub-imágenes o patches
289 de la imagen original. En este trabajo, se definen 10 tamaños de ventana de
290 igual alto y ancho, a saber 100, 200, 300, 400, 500, 600, 700, 800, 900 y 1000
291 píxeles, con un desplazamiento horizontal del 50% el ancho de la ventana y un
292 desplazamiento vertical del 50% el alto de la ventana, lo que produce una super-
293 posición del 50% entre parches contiguos. Estos valores se eligen sobre la base
294 del análisis de robustez del clasificador que presenta Pérez et al. (2017) para
295 la geometría de la ventana. Este análisis muestra que el clasificador (explicado
296 en la sección 2.3.3) es robusto para los patches que contienen al menos 60% de
297 los píxeles de una yema, y estos deben cubrir al menos el 20% del patch. Si
298 consideramos los casos extremos, i.e. el diámetro de yema más pequeño 100px y
299 el más grande 1600px, tamaños de ventana de 100px y 1000px podrían contener
300 al menos el 60

301 El segundo paso de este enfoque consiste en determinar si un patch es de
302 clase yema o no-yema. El clasificador de Pérez et al. (2017) toma los patches

303 producidos por el sliding windows y para cada uno realiza las siguiente opera-
304 ciones: (i) computa features visuales de bajo nivel mediante el algoritmo *Scale*
305 *Invariant Feature Transform* (SIFT) [Lowe \(2004\)](#); (ii) construye un descriptor
306 de alto nivel para cada patch empleando el algoritmo *Bag of Features* (BoF)
307 [Csurka et al. \(2004\)](#) sobre los features SIFT del paso anterior; y (iii) determina
308 la clase de cada patch usando el descriptor BoF sobre un clasificador construido
309 mediante el algoritmo *Support Vectors Machine* [Vapnik \(2013\)](#). Los detalles del
310 entrenamiento de este clasificador se posponen hasta la sección 2.3.3 (Entre-
311 namiento SW).

312 Finalmente, el tercer paso del enfoque consiste en construir la máscara bina-
313 ria donde se encuentran etiquetados los píxeles que pertenecen a la clase yema
314 y no-yema. Esta máscara es construida a través de un esquema de votación
315 donde cada píxel suma un voto por cada patch que lo contiene clasificado como
316 yema, el cual podría ser de un máximo de 4 para algunos píxeles debido a que el
317 deslizamiento propuesto entre patches presenta solapamiento tanto horizontal
318 como vertical. Luego, se establece un umbral de votos mínimos ν que puede
319 tomar los valores del 1 al 4, de tal manera que los píxeles con una cantidad de
320 votos igual o mayor a ν son clasificados como yema, caso contrario se clasifican
321 como no-yema.

322 2.3. Entrenamiento de los modelos

323 En esta sección se dan los detalles del proceso de entrenamiento para cada
324 enfoque. Para poder contrastar ambos enfoques ambos se han diseñado para
325 recibir el mismo tipo de entrada, i.e. una imagen de una escena vitícola, y
326 para producir las mismas salidas, i.e. una máscara binaria del mismo tamaño
327 que la imagen original cuyos píxeles positivos representan los píxeles del tipo
328 yema, junto a las coordenadas (X,Y) de la localización de estas yemas. Esto
329 permite entrenar ambos con la misma colección de imágenes, que se describe
330 en el siguiente apartado, seguida de los detalles de entrenamiento específicos de
331 cada modelo.

332 2.3.1. Colección de imágenes

333 La colección de imágenes utilizada en este estudio es la misma colección
334 utilizada originalmente en [Pérez et al. \(2017\)](#), el cual se ha descargado de la URL

335 <http://dharma.frm.utn.edu.ar/vise/bc> indicada por los autores. La colección
336 completo está compuesta por 760 imágenes capturadas en condiciones natural
337 de campo, en invierno. Sin embargo en este trabajo solo se tomaron las 698
338 imágenes que contienen exactamente una yema. Cada imagen está acompañada
339 del ground truth, es decir una máscara con la segmentación manual de la yema.
340 Estas imágenes y sus máscaras fueron empleadas durante el entrenamiento y
341 evaluación de los modelos de detección. Para esto, la colección de imágenes se
342 separó en dos subconjuntos disjuntos: el *trainset* con el 80% de las imágenes y
343 el *testset* con el restante 20%. Esto resultó en un trainset de 558 imágenes y
344 un testset de 140 imágenes, ambos con sus respectivas máscaras ground truth.
345 De esta manera, los dos enfoques propuestos utilizan exactamente las mismas
346 558 imágenes durante el entrenamiento, y las mismas 140 imágenes durante la
347 evaluación.

348 2.3.2. Entrenamiento del enfoque FCN-MN.

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

356 Dado que la cantidad de imágenes en el trainset se considera escasa, para
357 lograr un entrenamiento robusto se emplearon dos técnicas ampliamente uti-
358 lizadas en la práctica: *transfer learning* Pan and Yang (2009) y *data augmen-*
359 *tation* Shorten and Khoshgoftaar (2019). El proceso de transfer learning se
360 realizó de la siguiente manera: (i) se implementa la red MobileNet original
361 propuesta en Howard et al. (2017); (ii) se inicializa la red con los parámetros
362 pre-entrenados sobre el dataset de benchmark ImageNet Kornblith et al. (2019);
363 (iii) se reemplaza la capa de clasificación multiclase de MobileNet por una capa
364 de clasificación binaria; (iv) se entrena la red como un clasificador de patches
365 yema y no-yema de forma análoga al entrenamiento de SVM, empleando el
366 trainset de patches balanceado luego de escalar todas sus imágenes a 224×224
367 píxeles; y (v) se toman los parámetros obtenidos en el paso anterior para ini-

368 cializar el encoder de nuestra FCN-MN, introducido en la sección 2.1. El proceso
369 de data augmentation se aplicó on the fly durante el entrenamiento, i.e. en la
370 medida que el proceso requería nuevas imágenes. Por cada imagen del traíset
371 se generaron 200 nuevas imágenes (111600 en total) aplicando simultáneamente
372 las siguientes siete operaciones, donde sus valores se tomaron de forma aleato-
373 ria con probabilidad uniforme: *rotación* de hasta 45° ; *traslación horizontal* de
374 hasta 40%; *traslación vertical* de hasta 40%; *shear* de hasta 10%; *Zoom* de hasta
375 30%; *flip horizontal*; y *flip vertical*.

376 Para el entrenamiento de las tres variantes de FCN-MN, la 8s, 16s, y 32s,
377 se requiere especificar el *método de optimización* y el valor de *dropout*, dos
378 parámetros típicamente definidos por el usuario. En este trabajo, los métodos
379 de optimización que se tuvieron en cuenta fueron: *Adam* con parámetros learn-
380 ing rate = 0.001, $\beta_1 = 0.9$ y $\beta_2 = 0.999$; *RMSProp* con parámetros learning
381 rate = 0.001 y $\rho = 0.9$; y *Stochastic Gradient Descent* con parámetros learn-
382 ing rate = 0.0001 y $\text{momentum} = 0.9$. Para el caso de dropout se consideraron
383 dos valores: 0.5 y 0.001. Estos valores fueron preseleccionados por experimenta-
384 ciones preliminares que no se discuten aquí. La mejor combinación de método de
385 optimización y dropout se determinó en tiempo de entrenamiento sobre un con-
386 junto de validación, utilizando el enfoque *4-fold cross validation* por 60 epochs
387 y batchsize igual a 4, variando sobre los tres métodos de optimización y los dos
388 valores de dropout. Los valores seleccionados fueron aquellos que maximizan
389 el promedio de la Jaccard's *Intersection-over-Union* (IoU) (Jaccard, 1912), en
390 los 4-folds sobre las 3 variantes, siendo IoU una medida de evaluación típica
391 en problemas de segmentación (ver sección 3.1.2). Para cada par de valores de
392 optimizer y dropout se calcula el promedio tomando el resultado de IoU de cada
393 uno de los cuatro folds sobre cada una de las tres variantes. Observamos en la
394 Tabla~2 que la combinación de parámetros con la que se alcanza mayor IoU
395 promedio es RMSProp con dropout de 0.001.

396 Finalmente se procedió a entrenar las 3 variantes con RMSProp como método
397 de optimización y un valor de dropout de 0.001 sobre el conjunto de entre-
398 namiento completo por 200 epochs y batchsize igual a 4.

	Mean IoU	
Optimizer	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 para cada par de valores de optimizer y dropout, calculado a partir del resultado de IoU de cada uno de los cuatro folds sobre cada una de las tres variantes. Promedio de IoU en los 4 folds sobre las 3 variantes para cada combinación de optimizador y dropout.

2.3.3. Entrenamiento enfoque SW

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

A partir de esta colección de parches, se creó un trainset de parches balanceado, i.e. con 558 patches de cada clase, donde los patches no-yema fueron tomados al azar entre miles de patches. El entrenamiento se realizó tal como se detalla en el pipeline propuesto en Pérez et al. (2017): (i) se extrajeron descriptores SIFT todos los patches del trainset; (ii) se aplicó BoF con tamaño de vocabulario igual a 25, dado que fue el modelo con mejores resultados según los autores; y (iii) se entrenó el clasificador SVM sobre los descriptores BoF de cada patch, empleando un kernel *Radial Basis Function*, donde el valor de

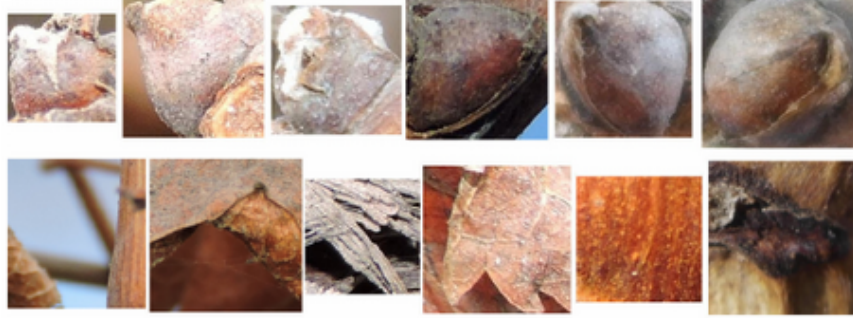


Figure 2: Collection of patches used in this work. The first and second rows correspond to bud patches and non-bud patches, respectively. Image extracted from Pérez et al. (2017).

los parámetros γ y C se estableció mediante un 5-fold cross-validation sobre los mismos rangos de valores, i.e. $\gamma = \{2^{-14}, 2^{-13}, \dots, 2^{-7}\}$ y $C = \{2^5, 2^6, \dots, 2^{14}\}$.

3. Experimental results

In this section we present a systematic evaluation of the quality our proposed procedure FCN-MN for bud detections quality, which, according to the discussion in the introduction, can be decomposed on the three aspects that impact on the relevant bud related variables listed in Table~?: *segmentation*, *correspondence identification*, and *localization*.

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

3.1. Performance metrics

3.1.1. Correspondence identification metrics

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

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

- 450 • **Correct Detection** (CD) is the best case, and counts all images in the
 451 test collection for which the detected binary mask presents a single con-
 452 nected component, and this connected component overlaps with the true
 453 bud of the image. This would correspond with a *true positive* situation.
- 454 • **Split** (S) occurs when there is more than one detection per bud, which
 455 happens when the mask contains multiple connected components, all of
 456 which overlaps the true bud. This metric counts the total number of
 457 images of the test collection whose detection is splitted.
- 458 • **False Alarm** (FA), is equivalent to a *false positive* situation, and corre-
 459 sponds to connected components not overlapping with the true bud. This
 460 measure counts the total number of such components over all images in
 461 the test collection.
- 462 • **Detection Failure** (DF), is equivalent to a *false negative* situation, when
 463 the detection mask presents no connected components. It counts one each
 464 image satisfying this condition.

465 All four of these cases are mutually exclusive, that is, no image can sat-
 466 isfy any two (or more) of these definitions simultaneously. To quantify the
 467 correspondence identification quality one could simply report these quantities
 468 counted over the test set, with the best case consisting in a CD value equal
 469 to the cardinality of this set. However, determining the overall correspondence
 470 identification quality from the analysis of 4 quantities can get rather compli-
 471 cated. One alternative is reporting the well-known precision and recall, denoted
 472 P_D and R_D and referred to as *detection-precision* and *detection-recall* to distin-
 473 guish them from the segmentation precision and recall defined later below. For

that, we have to address first the fact that we have two differing true positive counts: CD and S . We solve this by first counting as true positives not only the CD type of images, but the S ones, i.e., we count as one true positive any image with either a correct detection or a split case, resulting in:

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

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

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

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

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

3.1.2. Segmentation metrics

Correspondence identification metric, although informative, relies on the overlap between the detected and true buds, regardless of how minimal the overlap. This could miss several possible pixelwise detection errors, resulting in rather coarse comparisons between competing detection algorithms. For instance, a correct detection could present a very small overlap with the true bud, with many or even a majority of the true bud's pixels missing (i.e., several *false negatives* pixels), or could be erroneously reporting several pixels as bud pixels (i.e., several *false positives* pixels). Clearly, the best case scenario would be a case of correct detection with no false negative or positive pixels, that visually would correspond to a perfect overlap of the detected connected component and the true bud. Similarly, a pixel wise comparison of the masks could help assess the quality of the splits. The best split, for instance, would be one completely enclosed within the true mask, i.e., with none of its connected components presenting false positive pixels; while covering as much of the true bud mask as possible, i.e., presenting just enough false negatives to disconnect its components. Finally, a false alarm case, clearly presenting only false positive pixels, could be further assessed by the number of (false positive) pixels in its components.

500 The community has proposed several metrics to quantify segmentation er-
 501 rors. The most obvious ones are those that report the *fraction* of the whole
 502 image corresponding to *true positive* pixels, denoted TPF ; *false positive* pixels,
 503 denoted FPF ; and *false negative* pixels, denoted FNF . As for the correspon-
 504 dence identification metrics, one can simplify the analysis by considering the
 505 pixelwise precision and recall, denoted P_S and R_S and referred to as *segmenta-*
 506 *tion precision* and *segmentation recall*, defined formally as:

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

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

507 proposed independently by [Dice \(1945\)](#), thus usually referred to as the
 508 *Dice measure*. A common alternative to the Dice measure is the Jaccard's
 509 *intersection-over-union* ([Jaccard, 1912](#)), equivalent to $TPF / (TPF + FPF +$
 510 $FNF)$.

511 With these metrics, one could quantify the refinements discussed in the
 512 first paragraph above, by simply applying them, not to the whole mask, but to
 513 the individual correspondence identification cases. For instance, reporting the
 514 mean Dice measured over all correctly detected components; or, to refine the
 515 assessment of how bad is a split, one could report the mean Dice measure to all
 516 components of some split, or the mean Dice measure over all split components
 517 of all split images.

518 The case of false alarms is rather monotonous and not very informative, with
 519 zero precision and recall for all such components. Indeed, a pixelwise assessment
 520 of the gravity of a false alarm requires a quantification of the number of false
 521 positive pixels. One could simply consider the FPF , the fraction of all the image
 522 pixels that are false positives. Instead, we considered a normalization against the
 523 size of the bud to be more informative, resulting in the *normalized area*, denoted
 524 NA and defined formally as *the total area of the component corresponding to its*
 525 *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. Results

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: correspondence identification, 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 correspondence identification 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 segmentations 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

559 detected test image, i.e., each image with only one component overlapping the
560 true bud, including the corresponding standard deviation in parenthesis. For
561 the split group, the mean and standard deviation are computed over the mea-
562 sures computed only for the split images, i.e., over the images containing at
563 least two components overlapping the true bud. Here, the segmentation metrics
564 are computed over the union of all split components. For the false alarms we
565 reported the mean *normalized area*(NA), in this case computed individually for
566 each false alarm component, reporting at each cell its mean over all false alarm
567 components of all test images.

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

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

583 The table shows a clear improvement of FCN-MN over SW. For all metrics it
584 is the case that the best FCN-MN detector (bolded) improves (or ties) over the
585 best SW detector (bolded); represented in the table by underlying the one with
586 better metric; with the exception of the two segmentation recalls (for correctly
587 detected and splits) for which the SW case has a better (larger) mean, 98.8
588 versus 99.9 for correctly detected, and 74.7 versus 78.6 for the split case; and
589 the total split count S , with the best case for FCN-MN being 1 and 0 for the
590 best SW case. These improvements are not statistically significant, however,
591 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 correspondence identification 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~\ref{fig:detection-scatter-plot}, a graphical representation of a non-summarized version of the second and third columns of Table~\ref{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.

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~\ref{fig:4} the S values, corresponding to the fifth column of a (non-summarized version of) Table~\ref{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 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.

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: Correspondence identification, segmentation and localization metrics for the best FCN-MN and SW detection models. Each column shows two bolded cells, corresponding to the cell with better metric among all FCN-MN rows, and the cell with better metric among the SW rows. The larger of the two has been underlined, representing the best among all combined models, i.e., the best of the column.

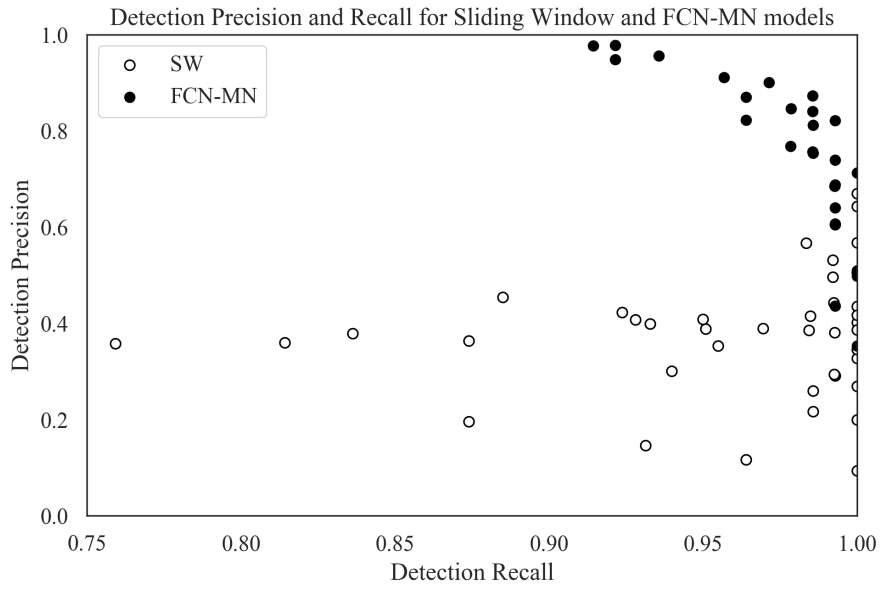


Figure 3: Precision-Recall scatterplots of the second and third columns of Table~3 discriminating the results for FCN-MN and SW with black and white dots, respectively. Each dot then represents the detection-precision and detection-recall computed over all images of the tests, for some particular configurations 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.

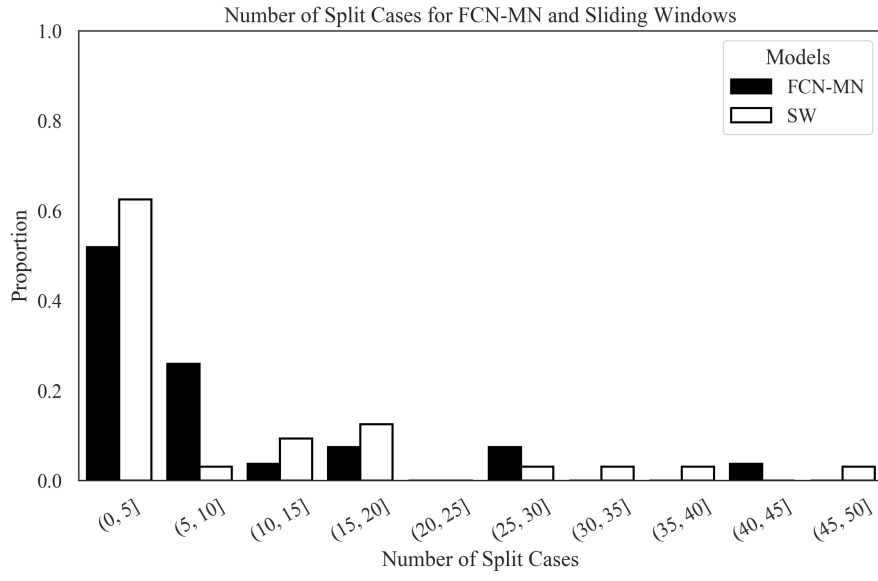


Figure 4: Histogram reporting the distribution of S for FCN-MN and SW in black and white bars, respectively. Each bar represents the proportion among all models (27 for FCN-MN and 40 for SW) that contains the number of splits indicated by the bin's label. For instance, the first (from left to right)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

3.2.2. Detailed analysis of segmentation metrics

As for the correspondence identification 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 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-MN 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 *NA* (over all false alarm components of all images) falls within the interval of the bin. The more concentrated to the left, the better is the algorithm, as this indicates that more detection models for that algorithm resulted in smaller *NA* (on average). One can observe the histogram for FCN-MN con-

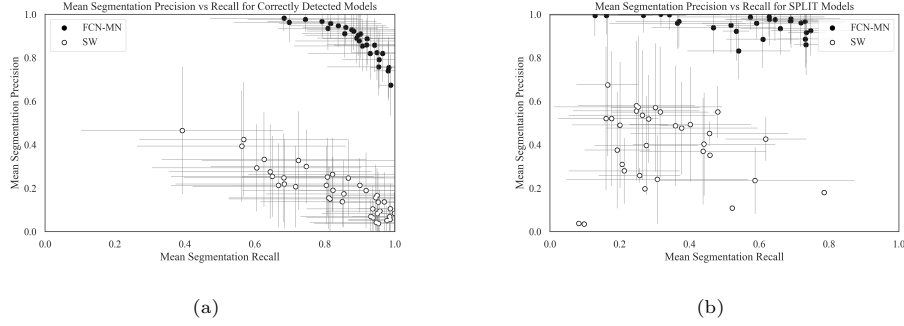


Figure 5: Segmentation Precision-Recall scatterplots reporting the results for FCN-MN and SW in black and white, respectively, with dots representing the average of segmentation precision and segmentation recall over all images in the test set (and bars representing standard deviations), with one dot 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.

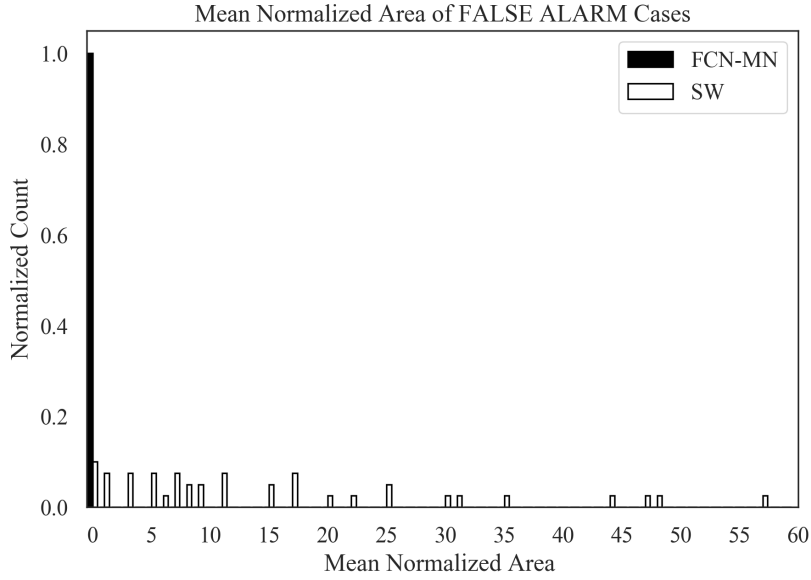


Figure 6: Two histograms of mean normalized area NA values reported for false alarms, one for the FCN-MN detection models (black) and one for the SW models (in white). Bars in the histogram represent the proportion of detection models whose mean NA (over all false alarm components of all images) falls within the interval of the bin.

655 siderably more concentrated at the left-most part of the histogram than that of
656 SW, with all FCN-MN concentrated in a single bar at the left-most interval of
657 $[0.0, 1.0)$. For SW the situation is rather different, with bars at intervals as far
658 to the right as $[57.0, 58.0)$, that is, detection models with areas as large as 58
659 times the area of the bud.

660 3.2.3. Detailed analysis of localization metrics

661 To conclude, we present in this subsection a graphical representation of the
662 localization results reported in Table~\ref{tab:TablaXX}, that is, the *normalized*
663 *distance*(ND) only for false alarms. This assumes that because they overlap
664 the true bud, correctly detected and split cases should be close enough to the
665 true bud to render it unnecessary any analysis on their distance. Instead, a false
666 alarm can be arbitrarily far from the true bud.

667 Figure~7 summarizes the ND values reported in the corresponding column
668 of the (non-summarized version) of Table~\ref{tab:TablaXX} in the form of two
669 histograms, one for FCN-MN (black) and one for SW (white). Bars in the
670 histogram represent the proportion of detection models (27 for FCN-MN and
671 40 for SW) whose mean ND (over all false alarm components of all images) falls
672 within the interval of the bin. The more concentrated to the left, the better is
673 the algorithm, as this indicates that more detection models for that algorithm
674 resulted in smaller ND (on average).

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

679 4. Discussion and Conclusions

680 En esta sección se discuten los resultados obtenidos por el enfoque propuesto
681 en el contexto del problema de detección de yemas de vid, su impacto como
682 herramienta para la medición de variables vitícolas de interés, y se destacan las
683 conclusiones más importantes y presentan los trabajos futuros.

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

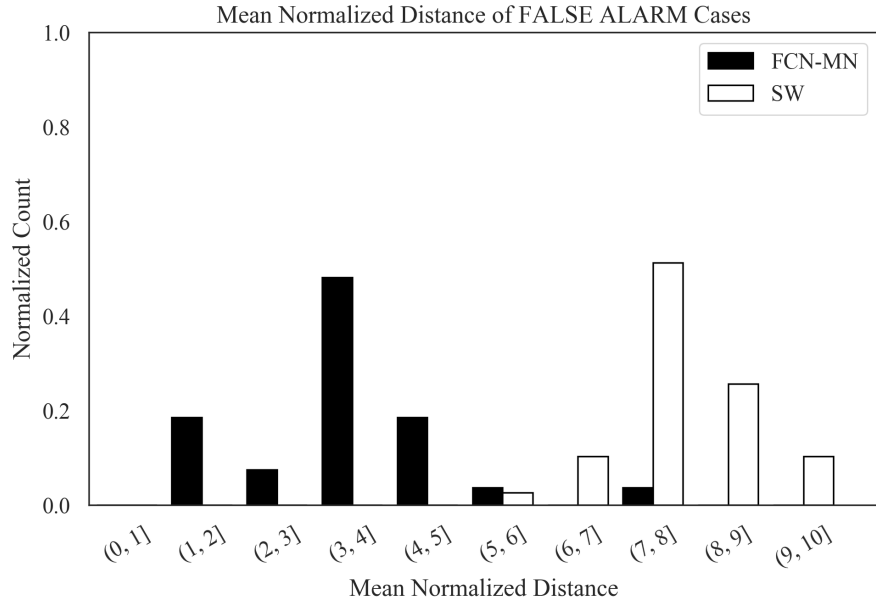


Figure 7: Two histograms of mean normalized distance ND values reported for false alarms, one for FCN-MN (black) and one for SW (white). Bars in the histogram represent the proportion of detection models (27 for FCN-MN and 40 for SW) whose mean ND (over all false alarm components of all images) falls within the interval of the bin. TODO:redactar

686 natural field conditions, in winter (i.e., with no leafs nor bunches), and con-
687 taining a maximum of one bud. The experimental results confirmed our main
688 hypothesis, that the detection quality achieved by FCN-MN improves over the
689 *sliding windows* detector (SW) in all three detection aspects: segmentation,
690 correspondence identification and localization. Being SW the best bud detector
691 known to these authors, one can conclude that FCN-MN is a strong contender
692 in the state-of-the-art for bud detectors.

693 But even improving over the state-of-the-art bud detectors one can still won-
694 der if it can address the main *quality* requirements of a practical measurement
695 of the bud related variables of Table~1.

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

719 $NA = 0.08$ and $ND = 1.1$, showing that these components are rather small,
 720 covering only an area that is 8% in size of the total area of a bud (on average),
 721 and distant to the true bud by only 1.1(0.65) diameters.

722 Based on these results, ¿what quality one should expect when the FCN-
 723 $MN_{16s}^{0.6}$ detector takes part in the measurement of the bud related variables?
 724 For brevity we discuss this for three variables from Table~1: *buds number*, *bud*
 725 *area*, and *internode length*.

726 El caso de *buds number*, por ejemplo, requiere to identify correspondences
 727 para las yemas de la escena, por lo que su calidad se verá impactada sólo
 728 por la métricas de detection precision and recall (95.6 and 93.6 respectively).
 729 Para evaluar este impacto asumimos que una planta tiene aproximadamente en
 730 promedio 240 yemas. El número de yemas por planta depende de muchos fac-
 731 tores, como ser sistema de conducción, varietal, tipo de tratamiento, época del
 732 año, entre otros, por lo que este valor se define a modo indicativo para lograr
 733 un análisis aproximado. Para este caso, una detection precision de 95.6 resul-
 734 taría en 11 yemas contadas en exceso por planta; mientras que la recall de 93.6
 735 resultaría en la omisión del conteo de 15 yemas. Además, este modelo produce
 736 3 splits con dos componentes cada uno, i.e. un error de conteo por exceso del
 737 3% sobre las 140 yemas del testset. Particularmente en este análisis significa
 738 que se contarían 6 nuevas yemas de más, dando un total de 17 yemas en exceso,
 739 prácticamente cancelando con el error de omisión. Pero además, estos errores
 740 podrían en la práctica caracterizarse estadísticamente, permitiendo corregir las
 741 mediciones hacia valores más certeros. A pesar de estos buenos resultados,
 742 nuestro enfoque presenta aún limitaciones prácticas para la medición del buds
 743 number debido a su imposibilidad de asociar en forma automática conteos de
 744 una misma yema en dos imágenes diferentes, dificultando la medición masiva
 745 del conteo de las yemas de una planta o parcela.

746 La segunda variable de interés considerada es *bud area*, donde, además de
 747 identificar las correspondencias para las yemas de una escena, es necesario seg-
 748 mentarla para estimar su área en píxeles. El análisis de correspondence identi-
 749 fication es análogo al del conteo de yemas, por lo que ahora se discuten sólo las
 750 métricas de segmentation. Del análisis desarrollado en los párrafos anteriores se
 751 puede concluir que los errores de segmentación por splits y false alarm tienen

un bajo impacto en los resultados generales, y por ende en la estimación de
bud area. Por otro lado, si se compensan los errores de segmentación para los
 correct detected (i.e. 11.8% of the buds pixels missing and 7.8% of the detected
 pixels covering the background), el error de estimación del área es solo de un
 4%. A efectos ilustrativos, vemos que este error es menor al error de precisión
 resultante de medir el área de una yema con un calibre. Si asumieramos que la
 forma de una yema se ajusta a una circunferencia, y que el diámetro típico de
 una yema es de 5 mm de diámetro, obtenemos un área de $19.63mm^2$. Siendo
 que un calibre tiene una precisión es $0.1mm$, el error de precisión del área sería
 de $\pm 1.7mm^2$, equivalente a un 8.6% del área total; un monto que duplica el
 error del 4% producido por nuestro detector FCN-MN. A está diferencia se le
 debe además sumar el error de la medición manual resultante de asumir una
 forma circular de la yema, aproximación innecesaria en el caso de FCN-MN.
 Al igual que para el caso del conteo, estos buenos resultados en la precisión
 de la medición se ven limitados para alcanzar un uso práctico de este tipo de
 medición al verse imposibilitado de asociar en forma automática mediciones de
 áreas de una misma yema en dos imágenes diferentes, dificultando la medición
 sistemática de esta variable para las yemas de una planta o parcela. Además,
 en este caso, las áreas obtenidas son en píxeles, necesitando ser convertidas a
 magnitudes de longitud o área.

Por último, consideremos el caso de la *internode length*, estimada por la
 distancia entre yemas de una misma rama (por la cercanía entre buds y nodes),
 la cual involucra las operaciones de correspondence identification y localización.
 De nuevo, el análisis de correspondence identification es análogo al del conteo de
 yemas, que en este caso resultará en el reporte de más de una distancia debido
 a la detección de más de una componente por yema. Entre estas distancias,
 entendemos que el peor caso puede darse entre los false alarms, siendo estos los
 más alejados de la true bud, y entre dos yemas ocurre cuando las false alarms
 están a distancia ND del lado más alejado de la otra yema. En promedio, $ND =$
 1.1, que de acuerdo al diámetro típico de las yemas de vid equivale a aprox.
 5mm, un valor muy menor a las distancias típicas de yemas de aproximadamente
 15cm, i.e., alrededor de un 6.6% de error en la estimación de la distancia entre
 buds/nodes. Una limitación de nuestro enfoque para alcanzar un uso práctico

785 de este tipo de medición es la posibilidad de determinar cuando dos yemas se
786 encuentran en la misma rama, lo que requiere conocer la estructura de la planta.
787 Además, con nuestro método, podría medirse solamente la distancia proyectada
788 en el plano de la imagen, la cual puede diferir arbitrariamente de la distancia
789 real en 3D.

790 Vemos que los errores de mayor impacto ocurren por el exceso u omisión de
791 connected components, con el error de exceso exacerbado por el hecho de asociar
792 buds detectadas con connected components individuales. Una mejora posible
793 para mitigar estos errores consistiría en aplicar algunos post-procesamientos.
794 Uno de ellos es el *spatial clustering* de los connected components que los agrupe
795 por cercanía. One could expect this to improve the results based on the small
796 areas of split and false alarm components. On one hand, due to the closeness
797 to the true bud of the false alarms (small ND), as well as the splits and cor-
798 rectly detected components (they overlap with it); and the fact that true buds
799 in real plants are typically tens or even hundreds of bud diameters apart, a
800 simple spatial clustering of the components would connect all these components
801 together as one single, and correct, bud detection. Second, due to their small
802 area, if clustered together, the false alarm components would only slightly re-
803 duce the segmentation precision. Otro posible post-procesamiento consistiría
804 en descartar connected components pequeños, por ejemplo, cuya área en píxeles
805 normalizada respecto al área total detectada (suma de las áreas de todos los con-
806 nected components) sea menor a cierto umbral. Podrían esperarse mejoras con
807 este post-procesamiento dado que los resultados en este trabajo muestran que
808 los false alarms presentan áreas pequeñas en relación al true bud. Por último,
809 podrían considerarse filtros de connected components basados en la estructura
810 de la planta, por ejemplo, descartando connected components que están lejos (o
811 no presentan overlap) con las ramas.

812 Also, one could consider in future works some improvements that overcome
813 the limitations for a practical use mentioned above: (i) no associations between
814 parts of plants of different images, (ii) distance and area measurements are in
815 pixels, (iii) only 2D geometry, (iv) lack of knowledge of the underlying plant
816 structure, and (v) need of images with no leaves.

817 One could consider extending to buds the work of Santos et al. (2020) that

addresses limitation (i) for grape bunches. Limitation (ii) could be easily addressed by adding to the visual scene some marker with known dimensions. This, however, requires such a marker in every image captured, a problem that could be overcome by first producing a calibrated 3D reconstruction of the scene, i.e., a 3D reconstruction calibrated with a single marker in one of its frames [Hartley and Zisserman \(2003\)](#); [Moons et al. \(2009\)](#). This way, every 2D image could be calibrated against the 3D model, omitting the need of a marker. In addition, a 3D reconstruction of the scene could address limitation (iii) by locating the detected buds in 3D space, following, for instance, the approach taken by [Díaz et al. \(2018\)](#). Finally, a solution to limitations (iv) and (v) would require an integrated solution involving the detection in 3D of branches and leaves, respectively.

Acknowledgments

This work was funded by the National Technological University (UTN), the National Council of Scientific and Technical Research (CONICET), Argentina, and the National Fund for Scientific and Technological Promotion (FONCyT), Argentina.

References

- Berenstein, R., Shahar, O.B., Shapiro, A., Edan, Y., 2010. Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer. *Intelligent Service Robotics* 3, 233–243.
- Bramley, R.G., 2009. Lessons from nearly 20 years of precision agriculture research, development, and adoption as a guide to its appropriate application. *Crop and Pasture Science* 60, 197–217.
- Chum, O., Zisserman, A., 2007. An exemplar model for learning object classes, in: *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE. pp. 1–8.
- Collins, C., Wang, X., Lesefko, S., De Bei, R., Fuentes, S., 2020. Effects of canopy management practices on grapevine bud fruitfulness. *OENO One* 54, 313–325.

848 Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C., 2004. Visual cat-
849 egorization with bags of keypoints, in: Workshop on statistical learning in
850 computer vision, ECCV, Prague. pp. 1–2.

851 Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detec-
852 tion, in: 2005 IEEE Computer Society Conference on Computer Vision and
853 Pattern Recognition (CVPR’05), pp. 886–893 vol. 1.

854 Diago, M.P., Correa, C., Millán, B., Barreiro, P., Valero, C., Tardaguila, J.,
855 2012. Grapevine yield and leaf area estimation using supervised classification
856 methodology on rgb images taken under field conditions. *Sensors* 12, 16988–
857 17006.

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

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

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

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

869 Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-
870 Gonzalez, P., Garcia-Rodriguez, J., 2018. A survey on deep learning tech-
871 niques for image and video semantic segmentation. *Applied Soft Computing*
872 70, 41–65.

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

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

879 Hartley, R., Zisserman, A., 2003. Multiple view geometry in computer vision.
880 Cambridge university press.

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

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

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

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

892 Institute, T.A.W.R., a. Viticare on Farm Trials - Manual 3.1: Measuring Fruit
893 Quality. 1 ed. The Australian Wine Research Institute. Accessed August
894 2020.

895 Institute, T.A.W.R., b. Viticare on Farm Trials - Manual 3.3: Vine Health. 1
896 ed. The Australian Wine Research Institute. Accessed August 2020.

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

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

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

- 905 Kliewer, W.M., Dokoozlian, N.K., 2005. Leaf area/crop weight ratios of
906 grapevines: influence on fruit composition and wine quality. *American Jour-
907 nal of Enology and Viticulture* 56, 170–181.
- 908 Kornblith, S., Shlens, J., Le, Q.V., 2019. Do better imagenet models trans-
909 fer better?, in: *Proceedings of the IEEE conference on computer vision and
910 pattern recognition*, pp. 2661–2671.
- 911 Lampert, C.H., Blaschko, M.B., Hofmann, T., 2008. Beyond sliding windows:
912 Object localization by efficient subwindow search, in: *2008 IEEE conference
913 on computer vision and pattern recognition*, IEEE. pp. 1–8.
- 914 Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian,
915 M., Van Der Laak, J.A., Van Ginneken, B., Sánchez, C.I., 2017. A survey on
916 deep learning in medical image analysis. *Medical image analysis* 42, 60–88.
- 917 Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for
918 semantic segmentation, in: *Proceedings of the IEEE conference on computer
919 vision and pattern recognition*, pp. 3431–3440.
- 920 Lorenz, D., Eichhorn, K., Bleiholder, H., Klose, R., Meier, U., Weber, E., 1995.
921 Growth stages of the grapevine: Phenological growth stages of the grapevine
922 (*vitis vinifera* l. ssp. *vinifera*)—codes and descriptions according to the ex-
923 tended bbch scale. *Australian Journal of Grape and Wine Research* 1, 100–
924 103.
- 925 Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints.
926 *International journal of computer vision* 60, 91–110.
- 927 Matese, A., Di Gennaro, S.F., 2015. Technology in precision viticulture: A state
928 of the art review. *International journal of wine research* 7, 69–81.
- 929 May, P., 2000. From bud to berry, with special reference to inflorescence and
930 bunch morphology in *vitis vinifera* l. *Australian Journal of Grape and Wine
931 Research* 6, 82–98.
- 932 Moons, T., Van Gool, L., Vergauwen, M., 2009. *3D Reconstruction from Mul-
933 tiple Images: Principles*. Now Publishers Inc.

934 Ning, C., Zhou, H., Song, Y., Tang, J., 2017. Inception single shot multibox
935 detector for object detection, in: 2017 IEEE International Conference on
936 Multimedia & Expo Workshops (ICMEW), IEEE. pp. 549–554.

937 Noyce, P.W., Steel, C.C., Harper, J.D., Wood, R.M., 2016. The basis of defolia-
938 tion effects on reproductive parameters in *vitis vinifera* l. cv. chardonnay lies
939 in the latent bud. *American Journal of Enology and Viticulture* 67, 199–205.

940 Nuske, S., Achar, S., Bates, T., Narasimhan, S., Singh, S., 2011. Yield estima-
941 tion in vineyards by visual grape detection, in: 2011 IEEE/RSJ International
942 Conference on Intelligent Robots and Systems, IEEE. pp. 2352–2358.

943 Oguz, I., Carass, A., Pham, D.L., Roy, S., Subbana, N., Calabresi, P.A., Yushke-
944 vich, P.A., Shinohara, R.T., Prince, J.L., 2017. Dice overlap measures for
945 objects of unknown number: application to lesion segmentation, in: Interna-
946 tional MICCAI Brainlesion Workshop, Springer. pp. 3–14.

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

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

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

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

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

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

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

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

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

971 Siam, M., Gamal, M., Abdel-Razek, M., Yogamani, S., Jagersand, M., 2018.
972 Rtseg: Real-time semantic segmentation comparative study, in: 2018 25th
973 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 1603–
974 1607.

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

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

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

984 Tarry, C., Wspanialy, P., Veres, M., Moussa, M., 2014. An integrated bud
985 detection and localization system for application in greenhouse automation,
986 in: 2014 Canadian Conference on Computer and Robot Vision, IEEE. pp.
987 344–348.

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

992 Vapnik, V., 2013. The nature of statistical learning theory. Springer science &
993 business media.

- 994 Wang, X., Han, T.X., Yan, S., 2009. An hog-lbp human detector with partial
995 occlusion handling, in: 2009 IEEE 12th international conference on computer
996 vision, IEEE. pp. 32–39.
- 997 Whalley, J., Shanmuganathan, S., 2013. Applications of image processing in
998 viticulture: A review .
- 999 Whelan, B., McBratney, A., Viscarra Rossel, R., 1996. Spatial prediction for
1000 precision agriculture, in: Proceedings of the Third International Conference
1001 on Precision Agriculture, Wiley Online Library. pp. 331–342.
- 1002 Xu, S., Xun, Y., Jia, T., Yang, Q., 2014. Detection method for the buds
1003 on winter vines based on computer vision, in: 2014 Seventh International
1004 Symposium on Computational Intelligence and Design, IEEE. pp. 44–48.
- 1005 Zhao, F., Rong, D., Liping, L., Chenlong, L., 2018. Research on stalk crops
1006 internodes and buds identification based on computer vision. MS&E 439,
1007 032080.